

Implementation of the Interagency Fuels Treatment Decision Support System



Final Report Prepared for
Joint Fire Science Program
Boise, ID

October 2012

This PDF document contains blank pages to accommodate two-sided printing.

Implementation of the Interagency Fuels Treatment Decision Support System

Final Report
STI-910901-5550-FR

Prepared by

Tami L. Haste
David J. Noha
Stacy A. Drury
Erin M. Banwell
Neil J. M. Wheeler
Michael D. Haderman

Sonoma Technology, Inc.
1455 N. McDowell Blvd., Suite D
Petaluma, CA 94954-6503
Ph 707.665.9900 | F 707.665.9800
sonomatech.com

Prepared for

Joint Fire Science Program
Bureau of Land Management
3833 S. Development Avenue
Boise, ID 83705

October 31, 2012

Acknowledgments

Many individuals have contributed to the Interagency Fuels Treatment Decision Support System (IFTDSS) software development effort. We greatly appreciate the support and guidance provided by John Cissel, the Joint Fire Science Program (JFSP) program manager. Erik Christiansen, Chair of the Fuels Management Committee (FMC), provided key help and support for analyzing the fuels treatment process that is a struggle for specialists from all agencies. Mike Rauscher, who has been the “boots on the ground” in representing the JFSP’s interests during the development of IFTDSS and who has skillfully facilitated the day-to-day communications between the JFSP and FMC, fuels treatment specialists, model developers, and the software development team. The JFSP Fuels Treatment Working Group (FTWG)—Michael Beasely, Dennis Dupuis, Mark Finney, Glen Gibson, Randi Jandt, David Peterson, Tessa Nicolet, and Brad Reed—has guided our work and provided initial critique and innovative ideas for the project. The JFSP Software Tools and Systems Study Advisory Committee—Pat Andrews, Nate Benson, Mike Hilbruner, Mike Hutt, David Peterson, Carol Saras, Paul Schlobohm, Shari Shetler, and Tim Swedberg—has kept the study headed in the right direction and made sure that all of the various components of a successful solution were considered.

We would like to specifically acknowledge the fuels treatment specialists who have agreed to participate in this effort to serve as the IFTDSS Test User Group—Brad Reed, Brenda Wilmore, Eric Miller, Gary Curcio, Gary Fildes, Gwen Lipp, Jim Roessler, Jon Wallace, Jonathan Olsen, Karen Folger, Mack McFarland, Nikia Hernandez, Perry Grissom, Randi Jandt, Randy Stiplin, Scott Weyenberg, Sean McEldery, Tessa Nicolet, Alison Forrestel, Brian Sorbel, Daniel Rasmussen, Dennis Fiore, Dennis Page, Glenn Gibson, Jeremy Spetter, Joshua Keown, Kim Kelly, Kyle Jacobson, Loretta Duke, Mike Uebel, Sam Amato, Anna Payne, Dave Pergolski, Jennifer Croft, Jeremy Bennet, John Washington, Ken Rodgers, Mathew Weldon, Paul Maday, Rance Marquez, Albert Savage, William Aney, Yanu Gallimore, Lauren Miller, Kristen Allison—who have provided, and will continue to provide, valuable feedback regarding the functionality and usability of the IFTDSS. We also thank the broader group of 49 field fuels treatment specialists who helped develop and refine the fuels treatment decision support process.

We would like to acknowledge our appreciation of the fire and fuels science and software development community—Eric Twombly, Mark Finney, Joe Scott, Alan Ager, Nick Crookston, Larry Gangi, Woodham Chung, Kurt Kruger—for their cooperation and feedback related to integrating data and software applications that will be a part of the IFTDSS. Thank you also to the Fire and Environmental Research Applications (FERA) team of Dave Petersen, Roger Ottmar, Susan Prichard, Paige Eagle, and Kjell Swedin for their support and assistance implementing new software modules into the IFTDSS. We would also like to acknowledge the managers of the Wildland Fire Decision Support System (WFDSS)—Tom Zimmerman, Rob Seli, Mitch Burgard and their development team—and the BlueSky Framework—Sim Larkin—for their willingness to work collaboratively to develop software systems that can communicate with one another to create efficiencies in the fire and fuels domain.

We would also like to thank the IT specialists who helped ensure that we have considered agency IT requirements—Brad Harwood, Laura Hill, John Gebhardt, and John Noneman.

Table of Contents

Section	Page
Acknowledgments	iii
List of Figures	vi
List of Tables	vii
Executive Summary	ES-1
 1. Introduction.....	 1-1
1.1 Software Tools and Systems Study	1-1
1.2 Community Development	1-2
1.3 Contents of This Report.....	1-3
 2. Software Design Approach	 2-1
2.1 Architecture Requirements	2-1
2.1.1 Strategic-Level Requirements.....	2-2
2.1.2 Community Requirements	2-2
2.1.3 Technical Requirements	2-3
2.1.4 General Software Requirements.....	2-4
2.1.5 Functional Requirements.....	2-4
2.1.6 Information Technology Requirements	2-5
2.1.7 Performance and Scalability Requirements	2-5
2.1.8 Information Technology Investment Requirements	2-6
2.2 Review of Existing Software Architecture Frameworks	2-6
2.2.1 Background	2-6
2.2.2 Distributed Service Oriented Architectures	2-8
2.2.3 Benefits of the SOA Approach	2-12
2.2.4 SOA Systems Assessment	2-12
 3. Overview of the IFTDSS Software Architecture.....	 3-1
3.1 The IFTDSS Web Application	3-2
3.1.1 Web User Interface.....	3-4
3.1.2 Collaboration Features	3-6
3.1.3 Help System	3-7
3.1.4 IFTDSS Conceptual Model and Database	3-10
3.1.5 Authentication Client.....	3-10
3.2 The Scientific Modeling Framework.....	3-12
3.2.1 WebUI Library.....	3-12
3.2.2 Executive	3-13
3.2.3 Scientific Data Storage	3-13
3.2.4 Acquisitor.....	3-13
3.2.5 Model Hosts.....	3-13
3.3 The Scientific Models	3-14
 4. Software Implementation Approach	 4-1
4.1 Agile Software Development Process.....	4-1
4.1.1 Create Product Backlog	4-2
4.1.2 Sprint Planning	4-2
4.1.3 Sprint Development	4-2
4.1.4 Sprint Review	4-3

Section	Page
4.1.5 Summary the IFTDSS Development Process	4-3
4.2 Test User and Developer Feedback	4-4
5. IFTDSS Functional Features.....	5-1
5.1 IFTDSS Workflows	5-2
5.1.1 Hazard Analysis Workflow	5-2
5.1.2 Risk Assessment Workflow.....	5-4
5.1.3 Fuels Treatment Workflow	5-6
5.1.4 Prescribed Burn Planning Workflow.....	5-9
5.1.5 Data Acquisition and Preparation.....	5-12
5.2 Developer-Designed Functionality	5-12
5.3 Direct Access	5-13
6. References	6-1
Appendix A: Response to Requirements	A-1
Appendix B: Hosting and Security	B-1

List of Figures

Figure	Page
ES-1. The three main components of the IFTDSS software integration framework.....	ES-5
1-1. Illustration of the IFTDSS stakeholder groups and the central role that the IFTDSS decision support team (transition community) will play in facilitating communication and collaboration.	1-3
2-1. A general diagram of the key SOA components.	2-9
2-2. Illustration of the concepts of situational application and mashup technology within an SOA environment.	2-11
3-1. IFTDSS component diagram.	3-3
3-2. The SMF provides a set of SMF-aware user interface components for use in web applications.	3-4
3-3. The IFTDSS searchable user list.	3-6
3-4. The IFTDSS project collaboration features, including (a) publishing and (b) searching for and acquiring published projects.....	3-7
3-5. Example of the mouse-over help provided.....	3-9
3-6. Example of the context-sensitive help pull-down menu.....	3-9
3-7. The IFTDSS Online User's Guide.....	3-10
3-8. The IFTDSS conceptual model database schema.....	3-11
4-1. The IFTDSS feedback process.....	4-5
5-1. Selecting how models will be accessed in a new project.	5-1
5-2. Overview of the hazard analysis workflow.	5-4
5-3. Overview of the risk assessment workflow.	5-6
5-4. Overview of the fuels treatment workflow.	5-8
5-5. Overview diagram of the prescribed burn workflow – IFTDSS Version 2.0.	5-11

List of Tables

Table	Page
5-1. Standard prescribed burn plan elements and the tools available in IFTDSS to help complete each element.	5-12
5-2. Separate calculations that can be performed in IFTDSS.....	5-14
5-3. Descriptions of models that IFTDSS calculations are based on.	5-15

Executive Summary

During the past two decades, a large number of data, software systems, and analysis tools have emerged in various research, modeling, and planning communities. The heterogeneity of available data, data formats, software systems, and ad hoc tools can fracture the awareness, access, and distribution of information and tools within a community. Consequently, analysts and decision makers are left with an assortment of analysis and modeling methods, as well as unconnected software systems in various stages of development.

In response to these issues, efforts are being made to develop agency-level (and interagency-level) information technology (IT) strategies and interoperability standards for integrating data and software services. The goal of these IT strategies is to reduce redundancy of software applications, to organize and integrate disparate data and models, and to help guide and streamline management and decision support processes.

This report describes a software integration framework that was developed to help organize and manage data and scientific models, and describes how it has been applied for the management of prescribed fire and vegetation.

ES-1. The Business Need

Over the past two decades, many software tools have been developed to help fuels specialists decide where, when, and how to manage vegetation (natural fuels) to reduce the risk of uncharacteristic wildfire. This proliferation of tools has come in response to various funding initiatives, with no guiding central governance or vision. All these tools can be effective in the right hands and for the appropriate purposes; however, the number of tools available, limited guidance on their use, and the time spent assembling data and learning each individual system has created frustration in the fuels planning community (Joint Fire Science Program, 2009).

Acting in concert with the interagency Fuels Management Committee (FMC), the Joint Fire Science Program (JFSP) initiated the Software Tools and Systems (STS) Study in 2007 to address the proliferation of unconnected and unmanaged modeling systems in the fire and fuels domain. A strategic assessment was performed (Palmquist, 2008) that led directly to development of a conceptual design and a software design for a service-oriented, framework architecture for fuels treatment planning (Funk et al., 2009). Under the guidance of an interagency team, these designs were developed into the Interagency Fuels Treatment Decision Support System (IFTDSS). In 2009, the JFSP funded development of a proof-of-concept version of the IFTDSS (Funk, 2010). A fully functional version of the IFTDSS is now under development.

While the IFTDSS software integration framework is specifically designed for fuels treatment planning, the framework approach serves as an example and stepping stone toward a larger “system-of-systems” vision. The vision is that the fire management community will access modeling, analysis, and reporting needs through a small number of interoperable software integration frameworks defined and organized by fire and fuels management business needs. For example, the BlueSky Framework (BlueSky) and Wildland Fire Decision Support

System (WFDSS) can also be considered software integration frameworks, each serving a different business need within the fire and fuels domain. Each of these larger software integration frameworks would eventually access a common virtual library of component computational models and tools. IFTDSS demonstrates the viability and value of this concept.

ES-2. Design Issues

Three major issues complicated the software architecture design of the IFTDSS: agency software administration and implementation restrictions; overlapping process implementations; and the desire for modular, reusable services.

ES-2.1 Agency Implementation Restrictions

Because of the multiple communities and agencies the IFTDSS must support, several implementation issues exist. Individual agencies have their own information technology policies and security restrictions, as well as software administration and installation requirements. In addition, the core user community has varying levels of technical skill.

ES-2.2 Overlapping Process Implementations

The heterogeneity and proliferation of data and software systems in the fire and fuels community resulted in overlapping science within different systems. For example, many of the software tools used to model fire behavior and fire effects are based on the same fundamental scientific algorithms. However, each software application varies in its implementation of the underlying algorithms and models; typically, bundling algorithms together and tightly coupling them with a graphical user interface. This approach makes it difficult and costly to update the underlying scientific models and to keep track of all of the different versions of the same models that exist.

The lack of modularity and clearly defined interface standards prevents changes in science from being implemented efficiently across multiple systems.

ES-2.3 Modular and Reusable Services

Developing and delivering standalone software applications is motivated by the desire to transform original scientific research results into decision support tools for managers. Software application development, deployment, maintenance, and user support operations are expensive and challenging to do well. The resources required to support a software application often compete with resources available to perform new research. The cost of developing a software application and the overhead required to support it are greater than simply coding the underlying mathematical algorithms into software models that can be shared and reused across many systems and applications.

ES-3. Design Approach

Our design approach consisted of five components:

1. stakeholder community engagement;
2. business needs;
3. architectural approach;
4. separation of functions; and
5. process-level science.

ES-3.1 Stakeholder Community Engagement

The literature of technology transition experiences shows that it is rarely sufficient to engage only the end-user community (Moore, 1991). Technology development teams allied with the early adopter end-users rarely have the resources or the staying power to move a new software technology from innovation to institutionalization on their own. The goal must be to design and deliver a “whole product,” which is the technology introduced plus everything else needed for the technology to be accepted and used. That is, it is a complete solution to the set of requirements developed (Forrester, 2007). To deliver the IFTDSS as a whole product, we needed the help and support of the broader stakeholder communities: governance, scientific model development, database stewardship, information technology and system maintenance, and fire and fuel management. In addition, an IFTDSS coordination team was needed to monitor and guide the software operations and the network of community stakeholders interacting with it.

ES-3.2 Business Needs

The system design focused on the most common business needs of the fuels planning community rather than on the underlying data and models used to support the business needs. Generally, fuels treatment planning involves assessing the existing landscape to identify areas where vegetation accumulation or environmental conditions pose a potential fire hazard, determining how to treat the vegetation on the landscape, and assessing the treatment effectiveness.

ES-3.3 Architectural Approach

Service Oriented Architecture (SOA) is a set of principles and methodologies for designing and developing software in the form of interoperable services. SOA-based frameworks provide a generic software architecture designed to support a collection of services, such as databases and software models, that are typically modular and can be reused for other applications. SOA has well-defined software and data interfaces, facilitates the integration of new and legacy software applications, and facilitates interoperability with other systems.

ES-3.4 Separation of Functions

To provide flexibility and extensibility to the system, we separated the system into three main functional parts:

- a web-based user interface
- a scientific modeling framework
- scientific models and modules

With this approach, the scientist-developers can focus on the data and models and the system provides the flexibility to extend and customize the user interface for various types of users.

ES-3.5 Process-Level Science

To facilitate the implementation of process-level scientific models while making it possible for existing software to still operate within the IFTDSS, a three-tiered approach to science integration was used. With this approach, new process-level science modules can be incorporated within the system, existing software is implemented within a standardized interface wrapper, and external systems can be accessed via web services.

ES-4. Architecture

Figure ES-1 shows the three main components of the IFTDSS. The first component, the IFTDSS Web Application (the user interface), provides the user experience and includes

- online help and documentation
- model selection, connection, and input
- spatial data visualization and editing; and
- collaborative features.

The IFTDSS Web Application is written in Java and uses JavaScript.

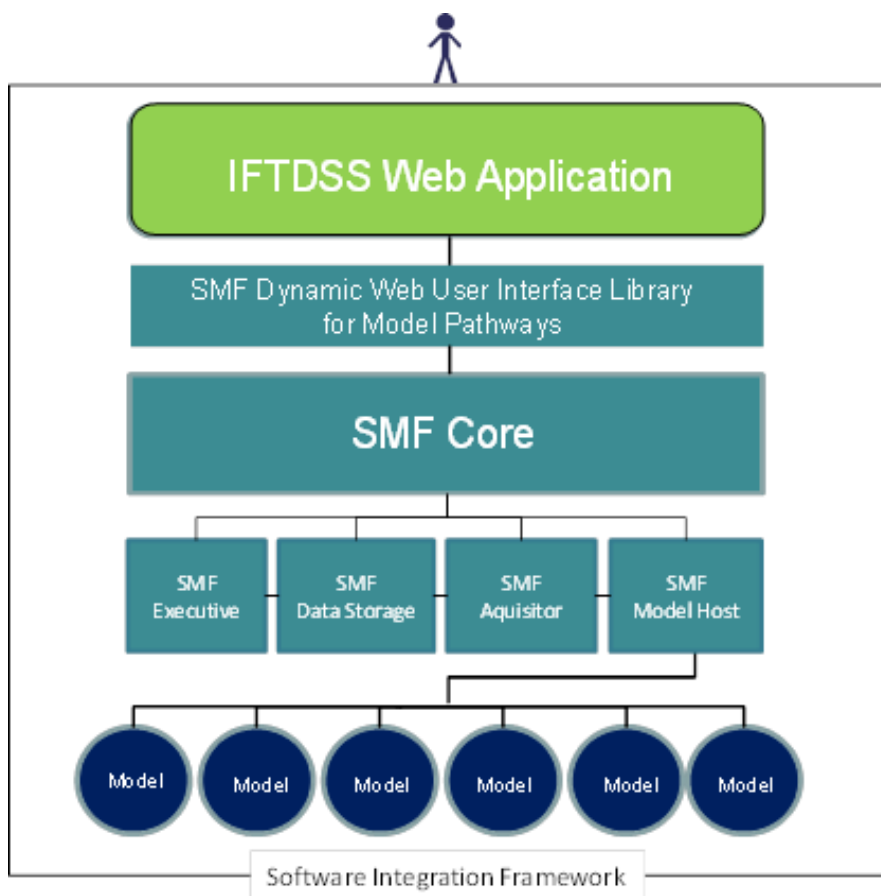


Figure ES-1. The three main components of the IFTDSS software integration framework.

The second component, the Scientific Modeling Framework (SMF), includes

- the SMF Core, which manages data flow and communication throughout the system;
- the SMF Executive, which is a registry for locating SMF service hosts and models;
- the SMF Data Storage server, which manages and stores multidimensional scientific data;
- the SMF Aquisitor, which provides a mechanism to import or upload data from external sources; and
- one or more SMF Model Hosts, which manage the execution of models.

The SMF Web User Interface (UI) library is an optional component of the SMF that provides a set of SMF-aware user interface components for use in web applications. The SMF Web UI's components and user-triggerable actions interact with SMF elements, such as data sets and models, while leaving the application with complete control over layout, data access, and model execution. The SMF is written almost entirely in Java; the SMF Data Storage uses netCDF for multidimensional data and supports other structured data used by the fuels treatment community.

The third component is the models. Models can be integrated into the IFTDSS by one of three methods:

1. direct integration into the system as a model subclass;
2. indirect integration by wrapping the model program using a custom interface, or model wrapper; or
3. through a web service connection.

While the direct integration method is the most efficient and provides the best control over process-level science, the other two methods provide needed support for legacy models and system interoperability capabilities.

The IFTDSS was created to be platform independent and is almost entirely built using open-source software solutions. The SMF and the SMF Web Application are written in Java and JavaScript; the Data Storage server is based on PostgreSQL; and the geospatial engine was built using Web Mapping Services and Open Layers.

ES-5. Development Process and Implementation

The IFTDSS is being developed using an agile software development process. The agile process is a group of software development methods based on iterative and incremental development, where requirements evolve through collaboration between end users, stakeholders, and the software development team. The agile process promotes adaptive planning and interim delivery of software functionality. The main benefit of the agile process is that it provides a mechanism to collect early feedback and encourages rapid and flexible response to change.

A functional prototype of the IFTDSS was completed in June 2010 and placed in service to obtain feedback from a test user group. Version 1.0 Beta was deployed in January 2012 to solicit feedback from the user community. A functionally complete version (Version 2.0) is delivered with this report.

ES-6. Functional Features

The modeling tools available in IFTDSS are grouped and are accessible in three ways:

1. by IFTDSS workflow,
2. by model developer, and
3. by individual models available in IFTDSS.

The IFTDSS workflows provide sets of tools for fuels treatment, prescribed burn planning, assessing fire hazard, and assessing potential risk from fire. The modeling tools are also grouped by model developer; that is, the tools are organized by the science teams that developed the models, and includes the model type and the outputs produced. The third way to access models within IFTDSS is to view a listing of all models.

ES-6.1 IFTDSS Workflows

Leading up to the development of IFTDSS, efforts were made to understand the decision support needs and workflow processes involved in fuels treatment planning and management. As a result of these efforts, the following four workflows have been identified and implemented in IFTDSS Version 2.0. Each workflow provides a logical, step-by-step process of using the various tools needed to perform the tasks of that workflow.

1. The **Hazard Analysis Workflow** is used to identify potentially hazardous areas across a landscape. The focus of this workflow is to identify areas across a landscape where fuels treatment analysis may be warranted based on potential fire hazard. IFTDSS provides tools that support this workflow.
2. The **Risk Assessment Workflow** provides a first-approximation probabilistic risk assessment for fuels treatment planning.
3. The **Fuels Treatment Workflow** (a) simulates fuels treatment placement in areas of high fire hazard within an area of interest, (b) simulates post-treatment influences on fire behavior and fire effects potentials, and (c) evaluates the temporal durability of fuels treatments; that is, how long, in years to decades, a treatment will continue to reduce adverse fire behavior and fire effects within an area of interest.
4. The **Prescribed Burn Planning Workflow** provides the information needed to plan and document a proposed prescribed fire. IFTDSS provides tools that support this workflow; with these tools, users can
 - calculate the probability of ignition from lightning or a firebrand
 - assess and calculate fire behavior
 - assess and plan fire containment
 - calculate fire effects
 - create a prescribed burn plan (including printing out a Word document with many of the required burn plan elements filled in by IFTDSS)

ES-6.2 Developer-Designed Functionality

IFTDSS supports the organization of tools by developer-designed workflows. These workflows can be a single calculation or a series of calculations implemented in the developer's original tool set or application. Currently, one developer's tool set has been incorporated into IFTDSS: the Fire and Environmental Research Applications (FERA) Team's Fire and Fuels Application (FFA). FERA is a USDA Forest Service research team focusing on fuels and fire and landscape ecology. IFTDSS supports several FERA tools:

- **Consume.** Predicts fuel consumption, pollutant emissions, and heat release based on a number of factors, including fuel loadings, fuel moisture, and other environmental factors.
- **Fuel Characteristic Classification System (FCCS).** Stores and classifies fuels data as fuelbeds, calculates physical characteristics of fuels based on fuelbed data, and calculates fire potentials based on the intrinsic properties of fuels.

- **Fire Emission Production Simulator (FEPS).** Predicts fuel consumption, emission rates, and heat release characteristics of prescribed burns and wildland fires. Total burn consumption values are distributed over the life of the burn to generate hourly emission and release information.
- **Digital Photo Series (DPS).** Users can link to the DPS from within IFTDSS to obtain fuel loading information for a selected situation to replace default values.

ES-6.3 Direct Access

IFTDSS provides direct access to a range of tools through standalone user interfaces to assist users who wish to perform calculations with a single tool instead of going through an entire workflow process. There are 71 individual calculations that can be performed in IFTDSS Version 2.0.

ES-7. Conclusions

The JFSP's vision for the IFTDSS extends well beyond fuels treatment planning. There are many other areas of the environmental sciences that could benefit from the IFTDSS's design for model integration, visualization, and system interoperability. Therefore, the IFTDSS SMF was designed to be generic so it can be applicable to any scientific discipline, and the IFTDSS application, which is database driven, can be easily customized. Further, the IFTDSS's SOA facilitates access to authoritative systems that are external to a DSS.

In many ways the issues and challenges faced by the fuels treatment planning community parallel those of other research and planning communities. Therefore, the approaches to model integration and the software tools and framework used in the IFTDSS may be transferable to the integration of process-level science in essentially any other field that relies on the organization, integration, and synthesis of information for use in decision-making.

Based on the IFTDSS design and development process, we have reached four main conclusions.

1. A DSS is more than a model. A model alone does not provide sufficient context to make decisions.
2. The development of an effective and sustainable DSS requires the participation of a broad community.
3. The scientific modeling framework and IFTDSS address long-standing issues with modularity and model interactions in the fuels treatment community.
4. The broader environmental science community faces many of the same challenges as the fuels treatment community and might benefit from lessons learned and engineering practices employed as a result of the STS study.

1. Introduction

A proliferation of software systems and analysis tools in the fire and fuels management domain over the past decade has made the task of managing and guiding the development of new tools and data sets increasingly challenging. Due to the heterogeneity of available data, data formats, software systems, ad hoc tools, and workflow processes in the fire and fuels community, awareness, access, and distribution of data and software tools has decentralized. As a result, fuels treatment specialists and decision makers are left with an assortment of unconnected systems in various stages of development and little guidance with respect to the strengths and weaknesses of these systems.

1.1 Software Tools and Systems Study

Through formal and informal interactions with its partners and clients, the interagency Joint Fire Science Program (JFSP) became convinced that the need for an integrated software architecture framework to manage the many models and data sets is a pressing issue facing fire and fuels analysts and decision makers. Acting in concert with the National Interagency Fuels Coordination Group (NIFCG), the JFSP initiated the Software Tools and Systems (STS) study in 2007. In Phase I of the STS study, the JFSP funded the Carnegie Mellon Software Engineering Institute (SEI) to perform a strategic analysis of the problem. This analysis was completed in March 2008, and SEI submitted a written report to the JFSP. A key finding of the SEI study was that the fire and fuels management community would greatly benefit from a software platform and a systems architecture that support integration and collaboration.

Following Phase I of the STS study, in the spring of 2008, the JFSP initiated Phase II of the STS study with the objective of designing the software architecture for an interagency fuels treatment decision support system (IFTDSS) that would provide a framework for organizing and integrating the many data and applications that serve the fuels treatment community. The Interagency Fuels Treatment Work Group (IFTWG) was formed to help guide the Phase II project, and a team from Sonoma Technology, Inc. (STI) was commissioned to help.

At the onset of Phase II of the STS study, the JFSP and the IFTWG developed a vision and conceptual design for the IFTDSS (Joint Fire Science Program, 2008). To ensure that the vision and conceptual design are consistent with current fuels treatment planning practices and that the IFTDSS will support the needs of the fuels treatment community, the JFSP, the IFTWG, and STI worked collaboratively to assess the current practices and needs of the fuels treatment community. The product of that assessment was a technical memorandum documenting the activities and findings of the current practices and needs assessment (Funk et al., 2008).

In Phase III of the STS study, system design specifications were prepared and a functional prototype of the IFTDSS was developed as a Proof of Concept (POC). During this phase, the POC's functionality was developed and confirmed in cooperation with the fuels treatment specialist community, namely, the IFTWG and fire and fuels application developers.

Upon successful completion of the POC, the JFSP and the Fuels Management Committee (FMC) funded Phase IV of the study to design, develop, and implement a fully functional version of IFTDSS. This report documents the outcome of Phase IV and the implementation of IFTDSS Version 2.0.

1.2 Community Development

A major goal of the overall IFTDSS project is to develop a community of individuals from multiple agencies and organizations who can collaborate, exchange, and communicate science and information related to fuels treatment analysis and planning. Collaboration among the fire and fuels community is important to improving the science and understanding of fuels treatment planning and to keeping the data, software applications, and the IFTDSS updated to meet current and future needs. In addition, collaboration helps all agencies and organizations learn from each other about methods, challenges, and approaches for fuels treatment planning.

The STS study has been a collaborative effort and has involved the active participation of four key IFTDSS stakeholder groups. During Phase II of the STS study, relationships were established with representatives from each stakeholder group. These groups were actively involved throughout the design, implementation, testing, and evolution of the IFTDSS. The success of the IFTDSS depends on continued active collaboration and engagement with each of the stakeholder groups. The following describes the key stakeholder groups and their involvement in the STS study:

- Fuels Treatment Specialists (approximately 1,000 throughout the United States). A group of more than 40 of these fuels treatment specialists was recruited to review and provide feedback on the design and development of the IFTDSS.
- Scientific Collaborators (approximately 20 throughout the United States). Scientific collaborators are those individuals or teams who develop new scientific algorithms, software, and tools to aid the fuels treatment planning community. During Phase II of the STS study, relationships were established with a subset of fire and fuels scientists and model developers (i.e., ArcFuels, INFORMS, IFP-LANDFIRE, Starfire, and other independent software model developers). The IFTDSS development team worked with scientific collaborators to ensure that analysis methods, models, and tools are properly implemented within the IFTDSS.
- Software Application and Data Provider(s). Institutional software application and data providers represent large-scale programs such as Landscape Fire and Resource Management Planning Tools (LANDFIRE) that provide analysis tools and data to the IFTDSS. This group should continue to be engaged in the IFTDSS process, and relationships should be further developed to foster inter-program collaboration.
- Management and Information Technology (IT) Administrators (representing the IT Investment process and hosting agency administrators). IT administrators will eventually administer and maintain the IFTDSS at a hosting agency. During Phase II of the STS study, the IT Investment team was engaged to help the IFTDSS architecture design team understand the interagency IT investment process requirements to ensure that the system will meet those requirements. This group should continue to be engaged throughout the implementation effort.

The involvement of these key stakeholder groups has been an integral part of the design process.

It is envisioned that the IFTDSS program will ultimately have a designated decision support team, or transition community, that can help facilitate communication and collaboration among the stakeholder groups. **Figure 1-1** is adapted from SEI's work in Phase I of the STS Study and illustrates the IFTDSS stakeholder groups and the central role that the transition community will play in facilitating communication and collaboration. Note that the stakeholder groups in this figure correspond to the stakeholder groups listed above: Fuels & Fire Operational Community = Fuels Treatment Specialists; SoS Platform Community = Scientific Collaborators; Models, Tools, Systems & Data Sets Community = Software Application and Data Providers; and Senior Management Community = Management and IT Administrators.

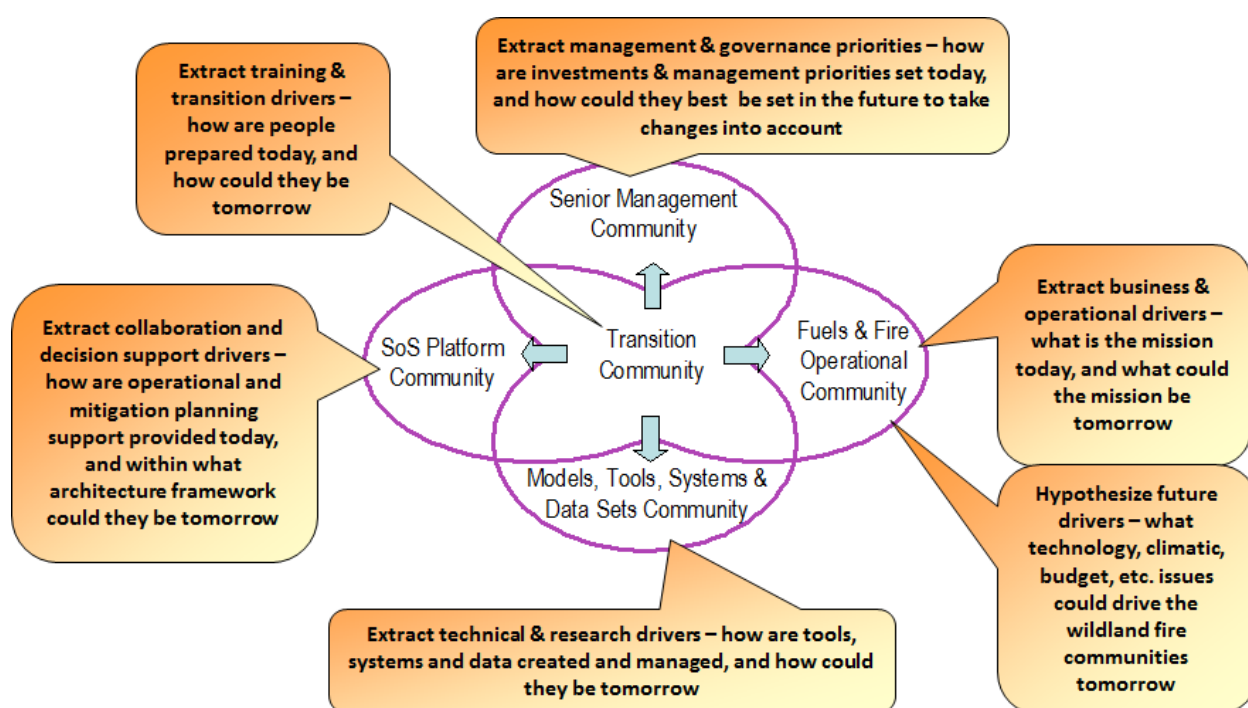


Figure 1-1. Illustration of the IFTDSS stakeholder groups and the central role that the IFTDSS decision support team (transition community) will play in facilitating communication and collaboration (adapted from Software Engineering Institute, 2008).

1.3 Contents of This Report

This report provides a description of the software design approach (Section 2), an overview of the IFTDSS software architecture (Section 3), a discussion of the IFTDSS development team's software implementation approach (Section 4), a description of the IFTDSS's functional features (Section 5), a list of references cited (Section 6), a summary of requirements and how they are addressed in IFTDSS (**Appendix A**), and, in **Appendix B**, a description of the IFTDSS 2.0 hardware and software configuration, as well as the development team's security practices.

2. Software Design Approach

Software design is a process of planning for a software solution to a particular problem. After the purpose and specifications of the software are determined, the plan for a solution can be developed. The IFTDSS software design addresses architectural, algorithm implementation, and low-level component issues.

2.1 Architecture Requirements

The IFTDSS architecture design was driven by the following objectives:

- Improve accessibility and organization of the assortment of data sources and fuels treatment planning tools that support the fire and fuels community.
- Simplify the fuels treatment planning decision support process and improve the overall quality of analysis and planning by making it easier to combine and reuse applications and by providing new opportunities for data analysis and collaboration.
- Facilitate scientific collaboration by providing a registration mechanism and tools that allow the integration of new software applications into the framework.
- Simplify project documentation and audit-trail tracking to support regulatory requirements.

Requirements define the intended purpose of a system under development. System requirements describe what the system will do and how it will be expected to perform. There are different types of requirements:

- **Strategic-level requirements** describe the key high-level requirements of the system and architecture.
- **Community requirements** describe tools, services, and standards that must be provided to the scientific community to achieve the integration goals.
- **Technical requirements** describe the high-level technical or platform issues.
- **General requirements** describe attributes of the system that generally apply to any complex software system.
- **Functional requirements** describe what the system must do and the functional attributes and characteristics that describe the functions that the architecture must support.
- **IT requirements** describe the interagency IT requirements.

The remainder of this section details the desired requirements for the IFTDSS. While it was recognized from the beginning that it might not be feasible to implement all of these requirements in the relatively short development period for IFTDSS Version 2.0, the goal was to meet as many of these requirements as possible.

2.1.1 Strategic-Level Requirements

The IFTDSS architecture will achieve its goals in several ways:

- Development of a unifying software framework to integrate applications.
- Centralization, organization, and management of fuels treatment data, software models, and analysis tools.
- Development of a registry system for new applications or updates to be distributed to the scientific community.
- Development of one or more scientific collaboration tools that can be used by application developers to assist in modifying the various software applications so they function within the new framework. These tools would make it easier for software developers to make their applications available to the IFTDSS and would support conformance verification, reusable software components, and other features to be determined.
- Specification of data standards, which will be supported by all integrated applications.
- Assistance and training for the scientific community, as necessary, to achieve the integration of the various applications.
- Training of fuels treatment specialists, as appropriate, via software user guides, integrated online help pages, and training programs.

2.1.2 Community Requirements

The variety of data and software applications used in the fire and fuels community differ in terms of manner of invocation, robustness, generality, types of modeling, execution platform, and in other ways. Integrating these applications poses both technical and structural challenges. This section discusses the structural challenges, or more precisely, the challenges of bringing together a diverse community of application developers and users to achieve a common goal.

Developers of individual applications are likely to have limited time, if any, to integrate their products with the proposed framework. Therefore, the following tools and guidance should be developed in the future to facilitate collaboration:

- Software tools to simplify and streamline the process of integrating new models into the IFTDSS.
- Technical assistance, including software application programming interface (API) documentation and email or phone support to application developers.
- Easy registration of components, and simplified delivery of applications and updates to users.
- Clear specifications for data standards, and specifications of required APIs that the software applications are expected to support.

In some cases, there may be key applications for which the original developer is not available or is unable to make modifications that are necessary for integration with the IFTDSS. In these cases, assuming the source code is available in the public domain and can be decoded, the IFTDSS development team may choose to integrate the application by

- obtaining source code, making necessary modifications, and integrating the application into the IFTDSS;
- “wrapping” the original software application in a wrapper application, which is itself integrated into the framework; and
- re-engineering the application in a way to allow it to integrate into the framework.

It is anticipated that in the early stages of the IFTDSS development, many of the software applications will require integration into the IFTDSS by one of the methods described above. Over time, when a sufficient number of key applications support the new framework, it is likely that there will be significant user pressure on remaining applications (and new applications being developed) such that voluntary support for the framework will be common.

The IFTDSS development team hopes that, by engaging and collaborating with the scientists and application developers (and users) of the various applications, the team will provide enough information that these groups will appreciate and support the value of integration from the IFTDSS framework. There is an obvious benefit to the community in general, and the expected value justifies the efforts that will be required. This concept is emerging as a similar type of integration for the smoke modeling community through effort on the BlueSky Framework, <http://www.getbluesky.org/>, and has proven effective in the air quality monitoring community through the U.S. Environmental Protection Agency’s (EPA) AIRNow program, <http://www.airnow.gov/>. The IFTDSS development team has drawn on these established collaboration and community building strategies when possible to benefit the IFTDSS community.

2.1.3 Technical Requirements

Existing fuels software applications perform a variety of functions and can be combined to perform complex simulations. The variety of existing applications developed with different goals and requirements, at different times, by different organizations, presents an integration challenge. Applications are written in different software languages, might run on different operating systems, have sometimes overlapping functionality, and require differing data formats.

The design of the IFTDSS software architecture was driven by the requirement to allow as many applications as possible to work together (including applications not yet developed) with a minimum of additional effort required by those application developers to support the framework. This objective was achieved by designing an architecture that is adaptable and generic enough to accommodate a broad variety of applications and functionality.

Government and state agencies often have rules and regulations regarding installation of new software on government computers and workstations. These regulations make it difficult to install software applications directly onto agency desktop computers. One of the requirements of the IFTDSS was that it could be run from a web browser by any desktop,

laptop, or workstation computer connected to the Internet; that way, users do not have to install any software on their local desktop computer. This feature implies that the system will have to be hosted on an accessible server and will need to be developed to function within the most commonly used web browsers (e.g., Internet Explorer and Firefox). See additional requirements in Section 2.1.6, Information Technology Requirements.

2.1.4 General Software Requirements

The IFTDSS architecture should have a long useful lifespan and supports the following software attributes during the architecture, design, and coding of the system:

- **Modularity** – each service (i.e., software applications, data sets, and tools) is developed in such a way that the component may be used independently of other software components. When such modules can be used independently of other program functions or combined with other modules, the user is benefited. It also benefits the system developers and maintainers, who can often reuse such modules in multiple applications.
- **Extensibility** – the system can be expanded over time to support the incorporation of new tools and data as they become available. This is a key requirement; as new applications are developed or old ones modified, they can be easily added to the framework.
- **Flexibility** – the system is flexible so users can customize data and model execution to fit their specific project analysis. This also has great benefits to the maintainers of a system. As platforms and requirements change, the software can be adapted.
- **Portability** – the system is easy to access and use from any standard desktop computer and does not require proprietary software or systems.
- **Ease of use** – the system is straightforward and practical to use through a well-designed interface. Specialized software training or programming skills are not required to run the system.
- **Maintainability** – the system should include clear structure and good quality technical documentation so system maintainers can easily maintain the system as features are added, platform requirements change, or defects are addressed.

2.1.5 Functional Requirements

The requirements listed here apply to the graphical user interface (GUI) application and to user experience or to the underlying framework. The system must

- support the decision support process, analysis steps, and software tools commonly used for fuels treatment planning;
- support visualization of spatial and tabular data, data editing, and user interaction at each processing step;
- provide data choices (i.e., standard treelist data, standard gridded data, and/or locally generated data);
- have data processing and transformation mechanisms to acquire or create and transform input data (i.e., the ability to combine vector and raster data formats; support

for vector-to-raster transformations and vice-versa). The system should contain mechanisms to streamline data preparation and processing;

- have a quality control, documentation, and audit trail mechanism to support regulatory requirements;
- provide guidance (i.e., submodel choices) based on geographic scale and the type of analysis being performed;
- be able to be stopped or started at any processing point;
- support analytical collaboration; that is, the system should provide a mechanism for fuels treatment analysts to publish and share methods and algorithms with other system users via a central system library;
- have a mechanism to perform sensitivity analyses;
- recognize user error and explain alternative actions; and
- support scientific collaboration; that is, the system must be able to incorporate new models and tools as they become available through an authorship and publishing mechanism.

2.1.6 Information Technology Requirements

The following requirements apply to the general IT operational requirements. The system

- must have an operation and maintenance plan and a long-term hosting agency with allocated servers, equipment, and maintenance staff;
- should be fully operational (24/7) and reliable;
- should be designed to function with high-speed Internet access (assumes users will have such access at a minimum);
- must support ArcGIS data formats and other commonly-used geographic information system (GIS) data formats; and
- must be designed for inter-operability with other decision support systems in the fire and fuels domain (i.e., BlueSky Framework and the Wildland Fire Decision Support System [WFDSS]).

2.1.7 Performance and Scalability Requirements

The following requirements apply to the overall performance and scalability of the IFTDSS. The system

- should be able to accommodate up to a maximum of 250 simultaneous users (about 25% of the total user community), running up to 500 computations per hour. These numbers can be increased with additional back-end hardware (i.e. a second server could be added to double those numbers);
- will ensure a response time of three seconds or less whenever a user invokes a command in the GUI. In cases of heavy load, or if the user is in the process of running an operation, the system response should be an indication that the command has been

recognized by the system and that the server is busy. In the latter case, some indication of progress, such as an active status bar, should be provided;

- should ensure that if a user repeatedly invokes an operation (because of issues with slow response time, for example) the repeated commands will not interfere with system stability (redundant commands will be ignored or deactivated to prevent this); and
- will store all intermediate calculations as they are produced by simulations, and any data the user enters will be stored as soon as it is received. In the case of a server crash, or if the system needs rebooting, all stored data will be available when the user logs back in. In addition, routine server backup processes will be employed to ensure that data are not lost in the case of a server failure.

2.1.8 Information Technology Investment Requirements

The National Wildfire Coordinating Group (NWCG) is in the process of developing a “cohesive interdepartmental-interagency method to manage the complex wildland fire Information Technology (IT) investment portfolio.” The IT Investment Process stewards IT projects from planning through development, operations, and finally to retirement.

As part of Phase II of the STS study, the IFTDSS team instituted a dialog with the creators of the IT Investment Process. Also, the IT Investment Process includes the planning and proposal phases, which the IFTDSS has already begun. However, it is expected that the IFTDSS development effort will serve as a pilot for the IT Investment Process, which will benefit both the IFTDSS project and the IT Investment Process development. For example, issues in the IFTDSS computer security plan might be identified, while gaps in the IT Investment Process will be discovered and rectified.

The IFTDSS development team has continued to engage and work with the NWCG and the IT Investment Process coordinators to ensure that the IFTDSS adheres to the Investment Process and the interagency IT requirements.

2.2 Review of Existing Software Architecture Frameworks

2.2.1 Background

The STS revealed several unique characteristics of the fire and fuels management community that warrant a platform and a software systems architecture that support organization and collaboration (Palmquist, 2008; Funk et al., 2008). These characteristics are described as follows:

- **A current lack of integrated and universally accepted analysis and planning methods.** The software architecture should provide a framework to integrate and organize the data and tools that fuels treatment specialists commonly use for analysis and planning, but should provide flexibility in how analysts use the tools for a particular problem or situation.
- **The use of expert judgment combined with models or model functions for fuels treatment planning.** It is critical that the software supports user interaction and

modularity, that is, users can independently select and use individual models or functions within the system.

- **The continuous development of new models and methods.** The software should be expandable and modular and should support the addition of new data, software models, and tools as they become available.
- **Computer administration issues.** Software installation and accessibility within many federal agencies is a concern. Therefore, the software must be accessible without requiring the installation of proprietary software and/or other resources that may be barriers to use.
- **A current lack of interagency collaboration.** There is currently a lack of interagency collaboration, although the agencies that perform fuels treatment planning face similar issues. The software should support open analytical collaboration by allowing users to publish and share their methods and algorithms within a system library.
- **Resource limitations.** Fuels treatment planners are often responsible for many tasks beyond fuels treatment planning and do not have the time or resources required to learn, and maintain familiarity with, dozens of software models and tools. Therefore, the software should provide analysis guidance and reporting tools to streamline fuels treatment decision-making.

A key, underlying characteristic of the fuels treatment planning community is the fact that, while there are many tools available for fuels treatment planning, no universally accepted analysis and planning approach exists. This lack of a coordinated approach is partly because of the geophysical and ecological complexities and specificities associated with fuels treatment planning and associated with the availability of vegetation data to support analysis. As a result, fuels treatment specialists usually employ customized data and/or analytical methods combined with expert judgment for decision making.

To address the characteristics of the fuels treatment planning community, the SEI study recommended that a platform and software architecture that supports interagency collaboration include the following key components (Palmquist, 2008):

- a software framework architecture that facilitates use and integration of data and scientific models, including a common user interface and shared data structures,
- the flexibility for users to select and compose their own data and chain of models to help address their specific analysis conditions,
- a clearly defined and articulated set of standards so that software developers can develop models and modules that will function within the software framework architecture, and
- a lifecycle management system with processes to set priorities for software system development, training, and retirement.

The challenge of organizing and managing the many data, software models, and tools within the fuels treatment community is not unique. Many businesses and research communities have faced similar challenges organizing information, resources, and work

processes to increase efficiency. As a result, a technological solution that has emerged over the past decade is the concept of Service Oriented Architecture (SOA). In simple terms, SOA facilitates the integration of disparate software systems by separating functions into distinct units, or services, that can be made accessible across a computer network so that users can combine and reuse individual services as needed (Erl, 2005). SOA facilitates the integration of data, new software systems, and legacy software systems to streamline work processes.

An example of SOA technology is online banking, where customers log in to a website hosted by their banking institution and manage their personal bank account(s) using a collection of individual services (i.e., bill pay, cash transfers, and account registers). Another example is the online tax preparation service, TurboTax®, where customers can prepare and manage their personal tax information online using a set of common tools, or services. SOA is a popular and widely used architectural approach for systems development and integration to support efficiency and collaboration within a community. A key recommendation resulting from the first phase of the STS study was that the fire and fuels treatment community would greatly benefit from an SOA solution (Palmquist, 2008).

2.2.2 Distributed Service Oriented Architectures

A universally accepted nomenclature for characterizing software systems architectures does not exist. Furthermore, key terminology related to SOAs, including “distributed” and “collaboration,” have different meanings to different authors. For the purpose of this report, we define “distributed computing” as computer systems working in parallel that are geographically or administratively separated. Some authors use the term “collaborator” to mean a person; others describe collaborators as computers and not the people using them. For the purpose of this report, we use the word “collaborator” to mean a person. We further distinguish between two types of collaborators: (1) system users who collaborate with one another within a problem space to share data, processing, and analysis methods, and (2) scientists who collaborate to improve the system by providing data, tools, and models to the system. We refer to these two types of collaborators as “analytical” and “structural,” respectively.

In the most general sense, distributed collaboration involves two or more geographically dispersed individuals working together to create a product by sharing and exchanging data, information, and knowledge. Collaborative environments are not software systems themselves, but they provide the framework to access and integrate data, models, and domain-specific tools to facilitate interdisciplinary collaboration.

SOA is becoming a popular and widely used approach for systems development and integration to support collaboration within a community. The benefit of effectively developed SOAs is a loose coupling of services with operating systems, programming languages, and other technological applications (Newcomer and Lomow, 2005). SOAs separate business or workflow functions into distinct units (services) that are made available across a network in a way that they can be combined and reused. With a central control system, the services can communicate with each other by passing data from one service to another, or by coordinating an activity between two or more services. SOA concepts are often viewed as a hybridization of distributed computing and modular programming (Erl, 2005). For the purpose of this report, we refer to SOAs that support dynamic Internet communication as “distributed SOAs.”

From a structural perspective, distributed SOA systems generally consist of two key components:

- A **central control system** that defines and tracks service registration and facilitates service transactions through the problem domain. The control system defines the requirements (i.e., interface specifications) for service interoperability.
- A published directory of registered **services**. Services are data and/or software module components that are platform-, programming language-, and operating system-independent.

Figure 2-1 provides a general diagram of the key SOA components.

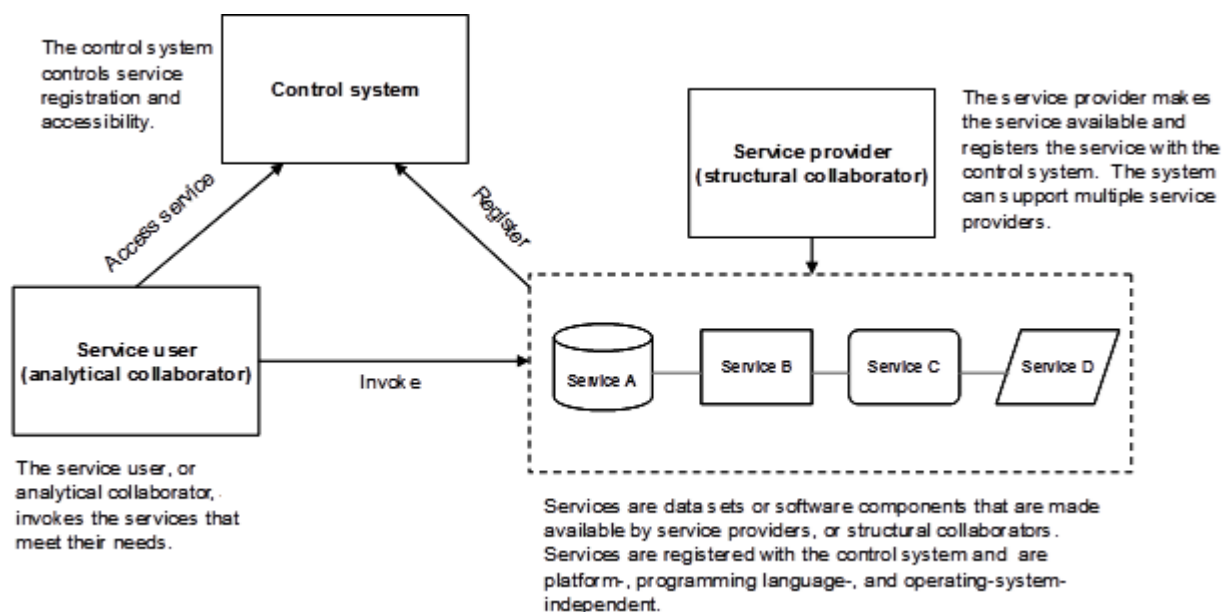


Figure 2-1. A general diagram of the key SOA components (adapted from Panda, 2005).

An SOA acts as a framework that brings together many disparate modules (services) and functions and supports their use and reuse in a controlled manner with fewer burdens on the user. For example, during fuels treatment analysis and planning, fuels treatment specialists often use fire behavior models such as FlamMap¹ combined with standard Microsoft Office products such as Microsoft Excel. Using this approach, the fuels treatment specialist must first prepare a vegetation data set, input that data into FlamMap, manually analyze the output(s) in Excel, and then prepare a fuels treatment prescription or burn plan. This approach can be time-consuming and requires extensive human manipulation of data, analysis expertise, and expert judgment at each step of the process. Several comprehensive standalone desktop software packages such as ArcFuels have been developed to aid fuels treatment specialists. These software systems provide tools that help integrate vegetation data with fire behavior models and provide analysis and visualization tools. While these software packages have proven to be

¹ <http://www.fire.org/index.php?option=content&task=category§ionid=2&id=9&Itemid=30>

extremely useful, they require a fair degree of knowledge of the system and software (e.g., ArcGIS), and generally offer specific fire behavior models (FlamMap in the case of ArcFuels) that are built into the system.

A well-designed and implemented SOA system would make it possible for the fuels treatment specialist to log in to a website, have the option to use pre-loaded vegetation data or to supply their own data, choose the fire behavior model that fits their particular analysis objectives (from a suite of available models), and execute a command that would tell the control system to perform the analysis in the specified sequence or path. The SOA system would then facilitate and automate the transfer of data and information from one step to the next to produce output information. The analyst would then view and manipulate the output data using a suite of available tools within the SOA. SOA technology can also facilitate multiple executions of the same path using a range of inputs to perform sensitivity analyses. A well-designed system would incorporate reporting tools that would help automate the development of documentation for the analyses performed.

A key feature of SOAs, and one that is critical for the fuels treatment domain, is the ability of users to define the specific services they wish to utilize and the sequence, or chain, of operation of these services. This concept is referred to as a situational application (SA). An SA is an application that has been created for a specific situational need, designed for and developed in collaboration with a specific social network or subset of system users (Watt, 2007).

A mashup is a type of SA that builds on services provided by different websites to create a new integrated experience. IBM introduced the term mashup to the software engineering world, and it has since become an accepted term (Watt, 2007). A very simple example of a mashup would be a University website (independent of the Google Maps website) that uses Google Maps functionality to provide an interactive map of the campus (e.g., <http://fullmeasure.co.uk/mashups/ecsitemap.htm>). Another example of a mashup is zillow.com, which integrates real estate tax information for a given geographic location (service A) with a map of the location (service B) so that users can view tax information for all real estate within a particular area on a map (new integrated experience). **Figure 2-2** expands on Figure 2-1 to illustrate the concept of mashups.

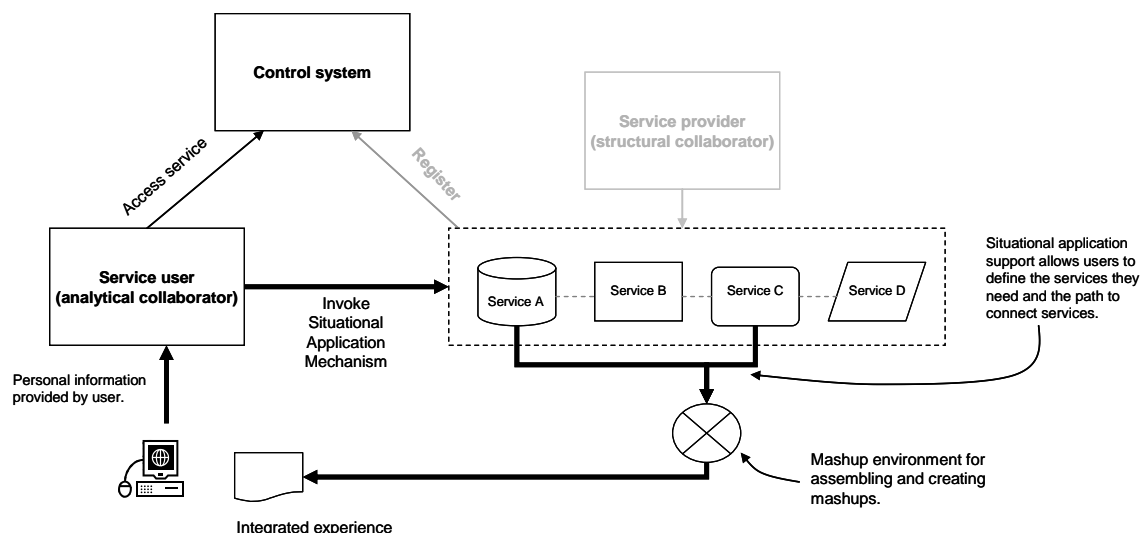


Figure 2-2. Illustration of the concepts of situational application and mashup technology within an SOA environment.

At least two SOA-based systems are already in use in the fire and fuels management community—the Wildland Fire Decision Support System (WFDSS) and the BlueSky Framework. The WFDSS was developed to support strategic and tactical decisions regarding real-time fire management, and the BlueSky Framework was developed to analyze and manage smoke impacts from fires.

The WFDSS combines desktop applications for fire modeling into a web-based system for easier data acquisition, and provides an easy way for fire managers and analysts to accurately document their decision making process. It organizes and manages its services to provide a decision support process and documentation system for all types of wildland fires. Because it is a web-based application, it facilitates analytical collaboration and sharing of analyses and reports across all levels of the federal wildland fire organization (http://wfdss.usgs.gov/wfdss/WFDSS_Home.shtml).

The BlueSky Framework combines desktop applications for fuel consumption and emissions simulations to produce estimates of emissions from fires. It consists of software framework programming code and accompanying models (services) that can be downloaded from a website and run on a local desktop machine. The Framework offers various model (service) choices at each step of a smoke impacts assessment. For example, the Framework can facilitate the use of the Emissions Production Model (EPM)² to estimate fuel consumption and emissions. Alternatively, users may opt to use the Consume 3.0³ and Fire Emission Production Simulator (FEPS)⁴ models. In addition, the architecture supports situational applications by making it possible for users to start or stop processing at intermediate steps or use different pathways for different types of fires. For example, a BlueSky Framework user may

² <http://www1.cira.colostate.edu/smoke/epm.htm>

³ <http://www.fs.fed.us/pnw/fera/research/smoke/consume/index.shtml>

opt to conclude processing after calculating emissions rather than continuing with further steps to model dispersion and ground-level smoke concentrations.

Both systems are demonstrating success in their specific user communities, and both systems currently use some of the same data sources and models used by the fuels treatment community. From a productivity perspective, these systems offer several key benefits to varying degrees.

- Integration and organization. Both systems offer an integrated and organized approach for performing analyses.
- Efficiency. Analysts no longer need to perform “overhead” tasks such as data formatting and intermediary data transfer between disparate models, which can be very time-consuming.
- Increased effectiveness. Support for analytical collaboration facilitates communication and sharing of information and analytical methods between collaborators. Support for structural collaboration provides a mechanism by which new or improved software models and tools can quickly be made available to the analyst community.
- Documentation support. Automated metadata generation and document tracking facilitate the preparation of documentation to meet formal reporting requirements.
- Reduces barriers to use associated with software administration. The systems organize and manage software tools so that tools don’t have to be installed and reside on a local desktop machine.

2.2.3 Benefits of the SOA Approach

As noted, the fire and fuels treatment community has access to a large number of software tools and data sources (Peterson et al., 2007). What it lacks is organization and integration of these resources into a single easily accessed system. SOAs are an ideal method of addressing this need. Using an SOA approach to organize and manage the software and data resources into services, and providing logical interactions among these resources (e.g. models), will more effectively facilitate fire and fuels treatment planning and decision-making. A well-designed SOA-based system would automate data and processing citation, facilitate analytic collaboration, and assure uniformity of analytical methods.

2.2.4 SOA Systems Assessment

The objectives of the JFSP’s effort are to gain an understanding of how similar communities have addressed the issues facing the fuels treatment community and to describe how a judiciously selected set of software architecture features could be used to effectively organize the fuels treatment software models and tools to support the work of the fire and fuels management community. To accomplish these objectives, seven existing distributed SOA systems were identified and assessed to (1) understand how they are used within their problem domains, (2) identify the system-specific functional features that are desirable to the fire and fuels community, and (3) identify the architectural features that support those functions. The

⁴ <http://www.fs.fed.us/pnw/fera/feps/>

intent in assessing the example systems is to gain insights that will prove useful in the selection and design of a software architecture to support the fuels treatment community.

The following seven systems each exhibit SOA properties. These systems were selected from both within and outside the fire and fuels domain. While none of these systems were directly applicable to the fuels treatment area, each system was examined to gain insights into how it is used to support analysis and/or decision-making within its specific problem domain:

- The **WFDSS** is a web-based system that streamlines and improves decision-making processes for resource and fire management response and planning. The WFDSS is governed by the USDA Forest Service (USFS), which also leads the responsibility for scientific development (http://wfdss.usgs.gov/wfdss/WFDSS_About.shtml).
- The **BlueSky Framework** is an open-source modeling platform that facilitates the use and interoperability of predictive models simulating the cumulative impacts of smoke on air quality from forest, agricultural, and range fires. The BlueSky Framework is governed by the BlueSky Consortium, with the USFS AirFire Team leading the responsibility for scientific development (<http://www.getbluesky.org>).
- The **Integrated Forest Resource Management System (INFORMS)** is a hybrid system that is controlled by a controller interface on a local PC but which runs system components across a network. The system facilitates fuels treatment planning activities across the USFS, and specifically helps support project-level National Environmental Policy Act (NEPA) analyses and landscape-level planning. INFORMS is supported by the USFS, with Colorado State University (Fort Collins) leading the responsibility for scientific development (http://www.fs.fed.us/psw/publications/documents/psw_gtr208en/psw_gtr208en_181-184_martinez.pdf).
- The **NOAA Harmful Algal Bloom Bulletin and Mapping System (HABMapS)** is a web-based interactive mapping system with a GIS that collects, stores, and displays various data layers used for detecting, monitoring, and tracking harmful algal blooms in the United States. The system was designed as a decision support tool for federal, state, and local resource and environmental managers and scientists. HABMapS is a collaborative effort supported by the National Oceanic and Atmospheric Administration (NOAA), NOAA CoastWatch, the National Ocean Service Center for Coastal Monitoring and Assessment, and the NOAA Coastal Services Center (http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20050237837_2005240380.pdf).
- The **U.S. Geological Survey Precipitation Runoff Modeling System (USGS-PRMS)** is a Unix-based, integrated modular modeling system that provides a framework to support the development, testing, evaluation, and dynamic integration of algorithms into models. The PRMS is used by river basin managers to simulate stream flow and by researchers to model hypothetical stream flow scenarios. With this system, users, or analytical collaborators, can create situational applications for specific areas and watersheds. And researchers, or structural collaborators, can develop and publish new models and subroutines. The PRMS is supported by the Center for Advanced Decision

Support for Water and Environmental Systems (CADSWES) at the University of Colorado and the USGS (<http://water.usgs.gov/software/PRMS/>).

- The **Federated Data System (DataFed)** is a web-services-based software framework architecture that facilitates collaboration among autonomous, distributed data providers and users in the air quality analysis community. DataFed facilitates registration of distributed data (on remote servers) into a centralized catalog and provides a basic set of tools for data exploration and analysis (aggregation, summary, visualization, etc.). All users and data providers have access to all data sources, and access is open to the public. The DataFed infrastructure was developed by the Center for Air Pollution Impacts and Trends Analysis (CAPTIA) at Washington University in St. Louis with funding from the National Science Foundation and the National Aeronautics and Space Administration (NASA) (<http://datafed.net>).
- The **Global Earth Observation System of Systems (GEOSS)** is a web-based software framework consisting of a portal, clearinghouse, and registry system for the exchange and dissemination of earth observation data. The purpose of GEOSS is to achieve comprehensive, coordinated, and sustained observations of the Earth system, to improve monitoring of the state of the Earth, increase understanding of Earth processes, and enhance prediction of the behavior of the Earth system. The GEOSS is a fully collaborative and open system in that all users and data providers have access to all data sources, and access is open to the public. The GEOSS infrastructure is supported by the EPA (<http://www.epa.gov/geoss/>).

All seven systems, to varying degrees, are distributed SOAs and provide data and/or tools for analytical collaboration.

3. Overview of the IFTDSS Software Architecture

The overall architectural design approach for the IFTDSS is based on the concept of SOA (as introduced in the previous section). SOA streamlines work processes and facilitates the integration of data, new and disparate software systems, legacy software systems, and applications by separating business processes into distinct units, or services, that can be made accessible so that users can easily access, combine, and reuse individual services as needed. (Erl, 2005). A business process is defined as a structured, measured set of activities designed to produce a specific service or product for a particular customer. In the context of this report, the customer is the fire and fuels treatment community.

A key benefit to the SOA approach is that it also makes it possible to combine and make available the same software models and processes through multiple GUIs so that different communities can use the system. SOA is a widely used architectural approach for addressing the same problem that the fire and fuels treatment community faces—the decentralization of disparate software applications written in various programming languages and of different vintages. A key recommendation resulting from Phase I of the STS study was that the fire and fuels treatment community would greatly benefit from an SOA solution (Palmquist, 2008).

The architecture presented here offers several benefits to users and developers of existing fire and fuels software applications. In addition to providing coordinated hosting of the diverse models used to assess fuels treatment options, this system delivers many advantages over independent execution of the models as currently required.

- The processes and services (i.e., the applications themselves) are separated from data loading and unloading activities.
- A small suite of data interchange standards were adopted.
- All processes and services were adapted to receive and produce data in standard formats.
- Models with aspatial processes, many of which are currently limited to a specific type of input format, were adapted to work with any combination of aspatial or spatial data formats.
- Resolution, coordinate system, and units incompatibilities were resolved automatically by a central control system.
- The system recognizes and exploits redundancies in the input data streams and in the modeled conditions. This feature allows optimal performance and storage efficiency.

Some of these key features result from architectural details that are distributed throughout the system. Others are focused in one or another component of the system. Many of the key features are embodied in the Executive, which is described in Section 3.2, The Scientific Modeling Framework. The Executive, in concert with other framework components, handles all or most of the overhead associated with formatting and passing data from one application to the next, thus freeing users from the burden of managing data formats and model processing steps in great detail.

Figure 3-1 illustrates at a high level the major components of the IFTDSS. At its core, the IFTDSS consists of three main architectural components: (1) the Scientific Modeling Framework (SMF), (2) the IFTDSS application, and (3) the scientific models that the system makes available. Two key characteristics of the IFTDSS architecture are worth noting: (1) the SMF is decoupled from the IFTDSS application (graphical user interface), making it a generic software framework for integrating scientific models and data; and (2) the scientific models and data can be integrated within the framework in a variety of ways, which makes it possible to accommodate existing applications in various configurations (e.g., desktop applications, web services, etc.). This section describes the IFTDSS architectural components.

Appendix A summarizes how these architectural components address the numerous requirements previously discussed in Section 2.1, Architecture Requirements. Appendix B describes the current hardware and software required to host these components and discusses future implementation and operational hosting issues.

3.1 The IFTDSS Web Application

The IFTDSS application contains the front-end web application and user experience elements. It provides users with a way to organize and share information about their projects and runs. It also contains general profile information for each user, as well as context-sensitive links to the online help and documentation that is specific to the IFTDSS. In summary, the IFTDSS Web Application is

- the IFTDSS GUI;
- available using standard web-browsers;
- loosely coupled with modeling framework; and
- written in Java using JavaScript.

The IFTDSS Web Application consists of a Web User Interface that users can use to:

- plan projects;
- execute workflows;
- visualize and edit spatial data;
- manage project data;
- collaborate with other users;
- obtain access to an online help system, training material, and technical documents.

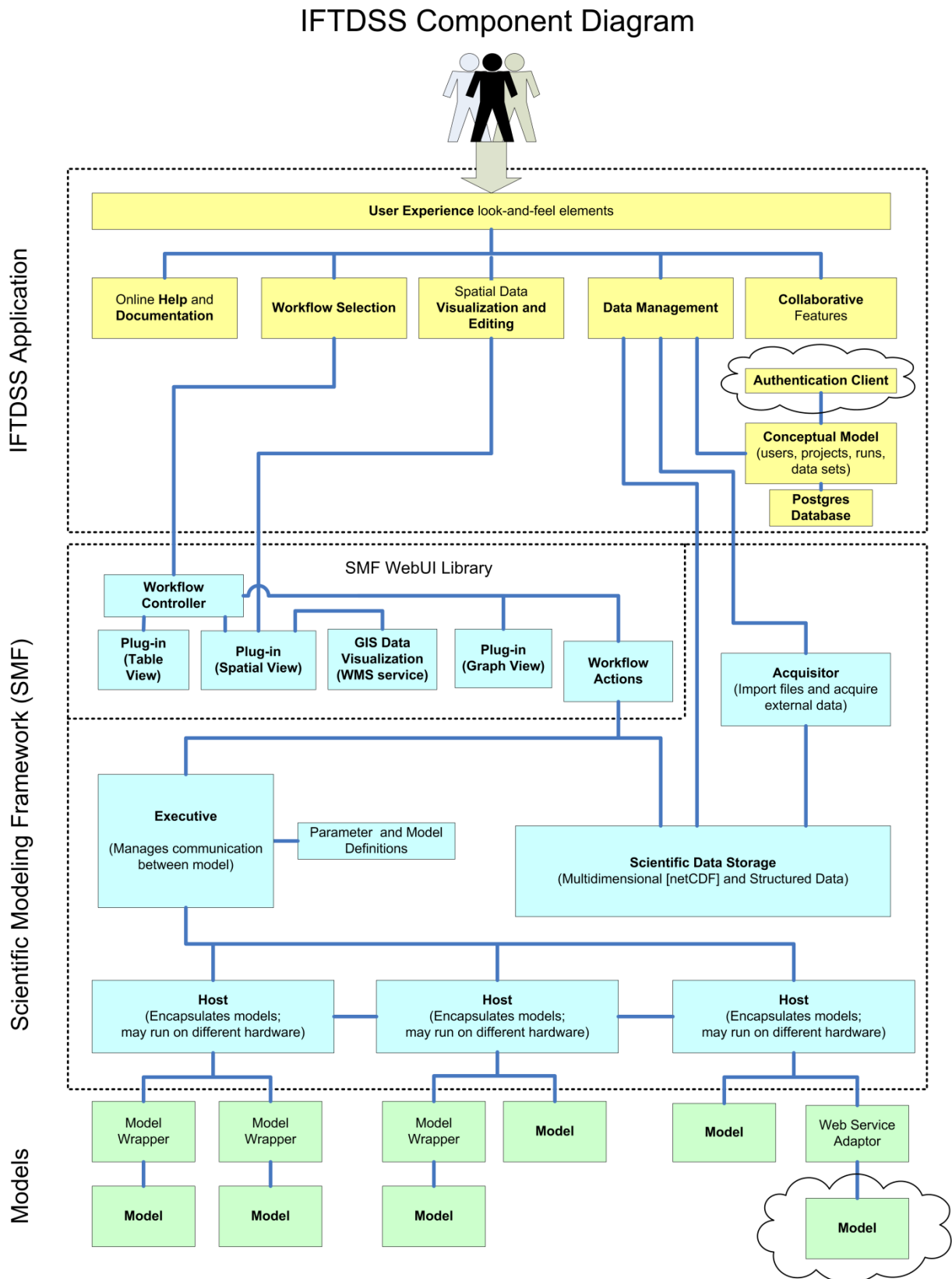


Figure 3-1. IFTDSS component diagram.

3.1.1 Web User Interface

The user experience and the manner in which a user interacts with the system are vitally important. The Web User Interface (WebUI) provides the mechanism by which the user interacts with the system and largely dictates the user experience. The WebUI makes it possible for users to interact with the IFTDSS. It controls the graphical look and feel of the IFTDSS, provides access to online help and documentation, and provides access to information about the IFTDSS and the scientific models within the system. The WebUI also serves the underlying system components and decision support processes and was customized to facilitate the fuels treatment decision support process.

Ultimately, the user experience is controlled by the application that is using the SMF. In the case of the IFTDSS application, we use the SMF WebUI Library, a component of the SMF that provides a set of SMF-aware user interface components for use in web applications (**Figure 3-2**). WebUI's interface components and user-triggerable actions interact with SMF elements such as data sets and models, while leaving the application with total control over layout, data access, and model execution.

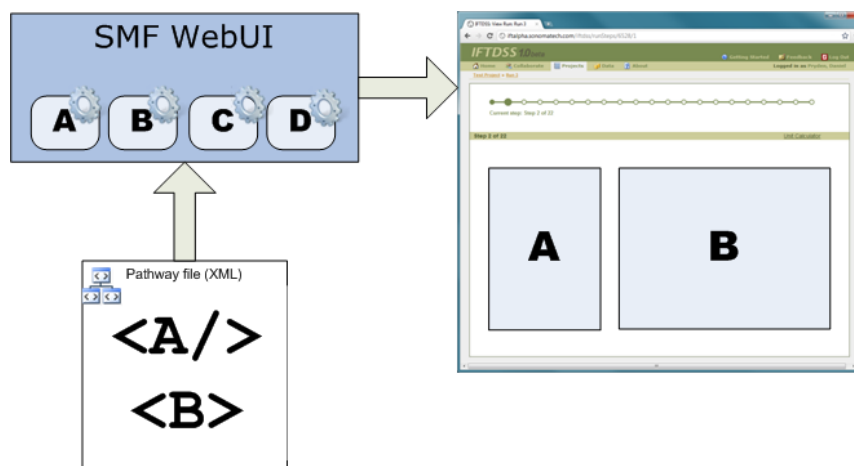


Figure 3-2. The SMF provides a set of SMF-aware user interface components for use in web applications.

Project Planning

The project planning layer is where the user sets up a new project analysis. All information related to a specific project will be input through the project planning tool, including: the project name; the analysis objectives and background information; the geographic location and extent of the study area; and the choice of vegetation data to be used (e.g., LANDFIRE, treelist data, local data). After a project has been defined, IFTDSS walks the user through a series of steps, starting with choosing a workflow or module and moving on from there through the process of selecting and analyzing data.

Workflows

WebUI components and actions are organized into scripted workflows called pathways, written in XML. A normal pathway might help the user prepare input data, run a single model, and view output data. Complex pathways might execute multiple models in sequence and crosswalk data between them. Simple pathways can even be used for data preparation or analysis, without running any model calculations at all. Pathways can also be used to prepare documents.

Pathways comprise one or more steps. Each step consists of one or more components that define the user interface appearing on a web page, as well as any actions that occur when the user completes that page (normally by choosing the **Next** button). In most pathways, the user must complete the steps in a specific order; in others, such as document preparation pathways, the user can navigate freely to any step.

The workflows have mechanisms that inventory and track parameters associated with a particular project, such as time intervals associated with treatments or disturbances and values at risk. It contains a catalog of approaches to solve particular types of problems; this catalog will evolve as users contribute to the analytical collaboration project library.

Spatial Data Visualization and Editing

The Spatial Data Visualization and Editing component provides users with tools to view and edit spatial (GIS) data. The geo-data visualization and editing tool is a GIS-based application that analysts can use to view, edit, perform spatial manipulations, stack map layers, and perform calculations on both spatial and aspatial data. It contains a GIS-based viewer as well as a tabular data viewer. It contains controls to quickly and easily perform unit conversions on specific data sets, create landscape (.LCP) files, and convert vector maps to raster maps and vice versa.

During an analysis, with the Spatial Data Visualization and Editing component, users can perform the functions that are required during an iterative treatment scenario analysis, including modifying vegetation conditions, designing and applying alternate treatment scenarios, and reviewing model output data. It also contains controls to handle many of the normally manual overhead steps associated with working with vegetation data, such as data conversions, calibration, imputation, and vegetation simulation.

Behind the scenes, this component automatically handles GIS-specific functions such as map projections. This component contains controls to export files in specified formats and view spreadsheets of data values (and map attribute data), and provides controls to export data in standardized formats that are compatible with other commonly used GIS systems, such as ArcMap and Google Earth.

Service Oriented Architecture

The SOA design of the IFTDSS makes it possible to create multiple WebUIs to serve specific purposes or user communities. For example, a Management WebUI could be created

that would provide a streamlined interface showing active projects, status, and results. A light-weight WebUI could also be developed for use on mobile devices in the field. The GUI described here represents the core interface that was developed to support fuels treatment specialists and scientific collaborators. The WebUI is web-based and operates in standard web-browsers.

3.1.2 Collaboration Features

The IFTDSS architecture provides access to several collaboration features. First-time users of IFTDSS complete a user profile that includes contact information (such as agency and location), and an optional bibliography, where expertise or research interests can be entered. A searchable list of IFTDSS users provides a mechanism for identifying and contacting others with similar interests (**Figure 3-3**). Clicking on a user's name in this list of users provides access to any information that the user has made available in their public profile. This information may include email address, mailing address, telephone number, and a narrative describing the user's background and interests.











IFTDSS Users						
Show <input type="text" value="10"/> entries		Search: <input type="text"/>				
Name	Organization	Geographic Region	City	State	Published Projects	
 account_test	Other	Other	Petaluma	CA	0	
 Andreu, Anne					0	
 Banwell, Erin	Private	Northern California	Petaluma	CA	0	
 Banwell, Erin	Bureau of Indian Affairs	Alaska	Petaluma	CA	1	
 Banwell, Erin	Private	Northern California	Petaluma	CA	2	
 Burgard, Mitch				MT	0	
 Cavallaro, Anthony	Other	Northern California	Petaluma	CA	0	
 Cissel, John			Boise	ID	0	
 Dove, Alex	US Fish and Wildlife Service	Rocky Mountain	Boulder	CO	0	
 Dove, Alexander	Tribal Forestry	Southern	Austin	TX	0	
<div> (all) (all) (all) (all) (all) </div>						
Showing 1 to 10 of 58 entries			First Previous 1 2 3 4 5 Next Last			

Figure 3-3. The IFTDSS searchable user list.

Once an IFTDSS project⁵ is established, it may be made available to other users through the project publishing feature as shown in **Figure 3-4a**. With this feature, users can share their work, learn from each other, and improve their analyses based on the experience of others in the fuels treatment community. Once a project is published, other users can search for and acquire a copy of the project using the interface shown in **Figure 3-4b**.

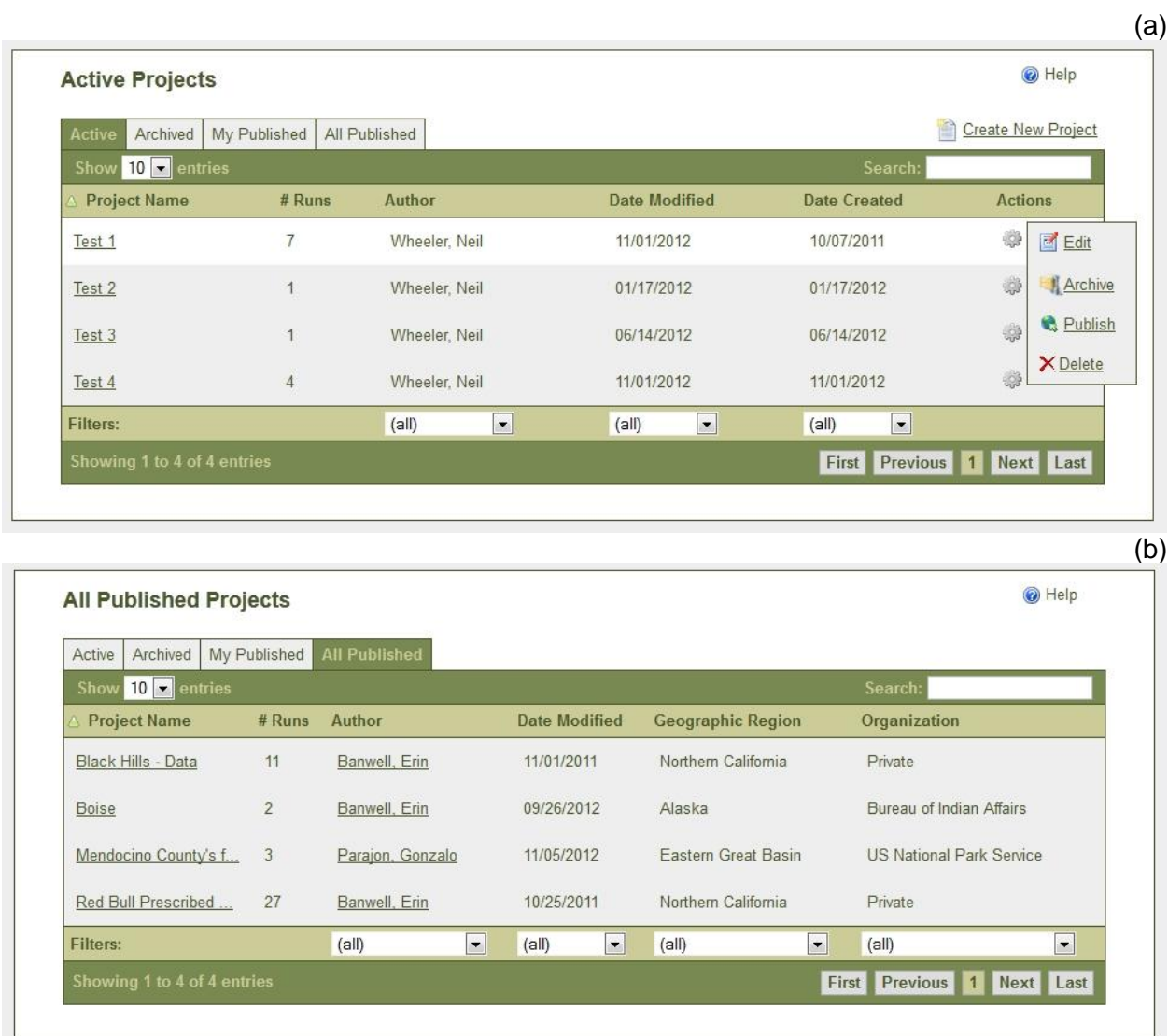


Figure 3-4. The IFTDSS project collaboration features, including (a) publishing and (b) searching for and acquiring published projects.

3.1.3 Help System

The IFTDSS help system provides three layers help to the user.

⁵ A project consists of one or more units that are organized and managed as a single endeavor.

1. **Mouse-Over Help.** Throughout the system, brief explanations of actions and inputs are provided when the mouse pointer is moved over the action or input name. **Figure 3-5** shows the mouse-over help provided when the pointer is over the 1-hr Fuel Moisture input.
2. **Context-Sensitive Help.** Once the user begins a workflow, a pull-down help menu is available that is specific to that workflow module (**Figure 3-6**). The menu items provide access to the online help system in a separate browser tab or window, and provide the specific information selected.
3. **Online User's Guide.** The entire User's Guide is available at any time in a separate browser tab or pop-up window by clicking the help icon at the top of each IFTDSS web page. The online User's Guide (**Figure 3-7**) provides fully indexed and searchable access to a variety of information to assist IFTDSS users.
 - Background information on the IFTDSS.
 - Tutorials and videos to assist new users.
 - Workflow-specific information and help on
 - Hazard analysis
 - Risk assessment
 - Fuels treatment
 - Prescribed burn planning
 - Fire and Environmental Research Application (FERA) Fire and Fuels Application (FFA) Tools
 - Reference material, including documents and links to information on the models and workflows implemented in the system.
 - Comparisons of IFTDSS results with those from the original models.
 - Glossary of terms used in IFTDSS.
 - User-specific bookmarks (Favorites).

The IFTDSS help system was developed and is maintained using MadCap Flare⁶. Flare allows integrated creation, management, and publication of content. Using Flare's import process, the help system developers were able to migrate a wide range of file types into an XML-based authoring environment. With XML, content is separate from formatting, which makes it possible for the authors to focus on content creation without the need for XML knowledge. Flare PDF and WebHelp outputs assist in making documentation more accessible to users who have visual and hearing impairments. It is compliant with standards such as Section 508 and Web Content Accessibility Guidelines.

⁶ See: <http://www.madcapsoftware.com/products/flare/overview.aspx>

The screenshot shows the 'Inputs' tab of the IFTDSS interface. At the top is a navigation bar with buttons: Configure, Inputs (selected), Review Landscape Data, Outputs, and Run Summary. Below this is a 'Model Information' section with a title bar 'Run 1 - Calculate fire behavior across a landscape (IFT-FlamMap)' and 'Help' and 'Tools' dropdown menus. The main content area is divided into 'Properties' and 'Fuel Moisture' sections. In the 'Fuel Moisture' section, there is a table with columns 'Parameter', 'Unit', and 'Simulation #1'. The '10-hr Fuel Moisture' row is highlighted, and a mouse-over tooltip is displayed over its input field. The tooltip text reads: 'Moisture content of 1-hour fuel (0 to 1/4" diameter dead woody fuel) is the mass of water within the fuel particle expressed as a percentage of the fuel particle's oven-dry mass. Range: 1 - 60 percent'.

Parameter	Unit	Simulation #1
1-hr Fuel Moisture	percent	6
10-hr Fuel Moisture	percent	
100-hr Fuel Moisture	percent	8
Live Herbaceous Fuel Moisture	percent	60
Live Woody Fuel Moisture	percent	90

Figure 3-5. Example of the mouse-over help provided.

The screenshot shows the same IFTDSS interface as Figure 3-5, but with the 'Help' dropdown menu open. The menu is context-sensitive and lists the following options: Module Information, Module Inputs, Two Crown Fire Calculation Options, Spatial Landscape Data, Fuel Models, and Module Outputs. The 'Inputs' tab is still selected, and the '10-hr Fuel Moisture' row in the table is highlighted.

Parameter	Unit	Simulation #1
1-hr Fuel Moisture	percent	6
10-hr Fuel Moisture	percent	7
100-hr Fuel Moisture	percent	8
Live Herbaceous Fuel Moisture	percent	60
Live Woody Fuel Moisture	percent	90

Figure 3-6. Example of the context-sensitive help pull-down menu.

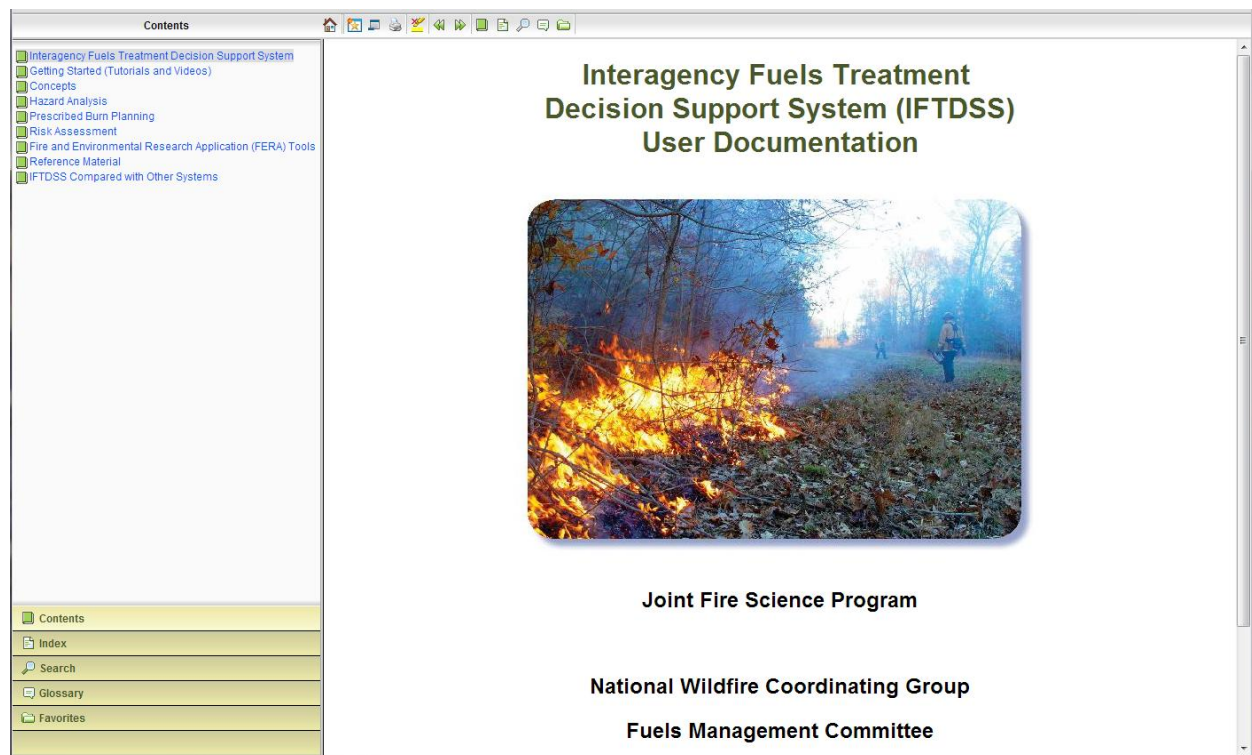


Figure 3-7. The IFTDSS Online User's Guide.

3.1.4 IFTDSS Conceptual Model and Database

The IFTDSS Conceptual Model component interacts with the Data Management UI and manages user-specific credentials, projects, run, and data sets. The Conceptual Model stores this information in a PostgreSQL (PostgreS) database. The schema for this PostgreS database is depicted in **Figure 3-8**.

3.1.5 Authentication Client

The authentication client is a separate service that verifies the credentials and access rights of IFTDSS users. This client may be implemented as a component of the IFTDSS Application locally or be a part of a larger, remote authentication system, such as the Fire and Aviation Management Web Applications web site (FAMWEB) that brings together a variety of applications, tools, and services related to interagency fire and aviation management managed by the NWCG and participating agencies. The authentication client is shown as a local client In Figure 3-1, with user credentials being stored in the IFTDSS Conceptual Model database.

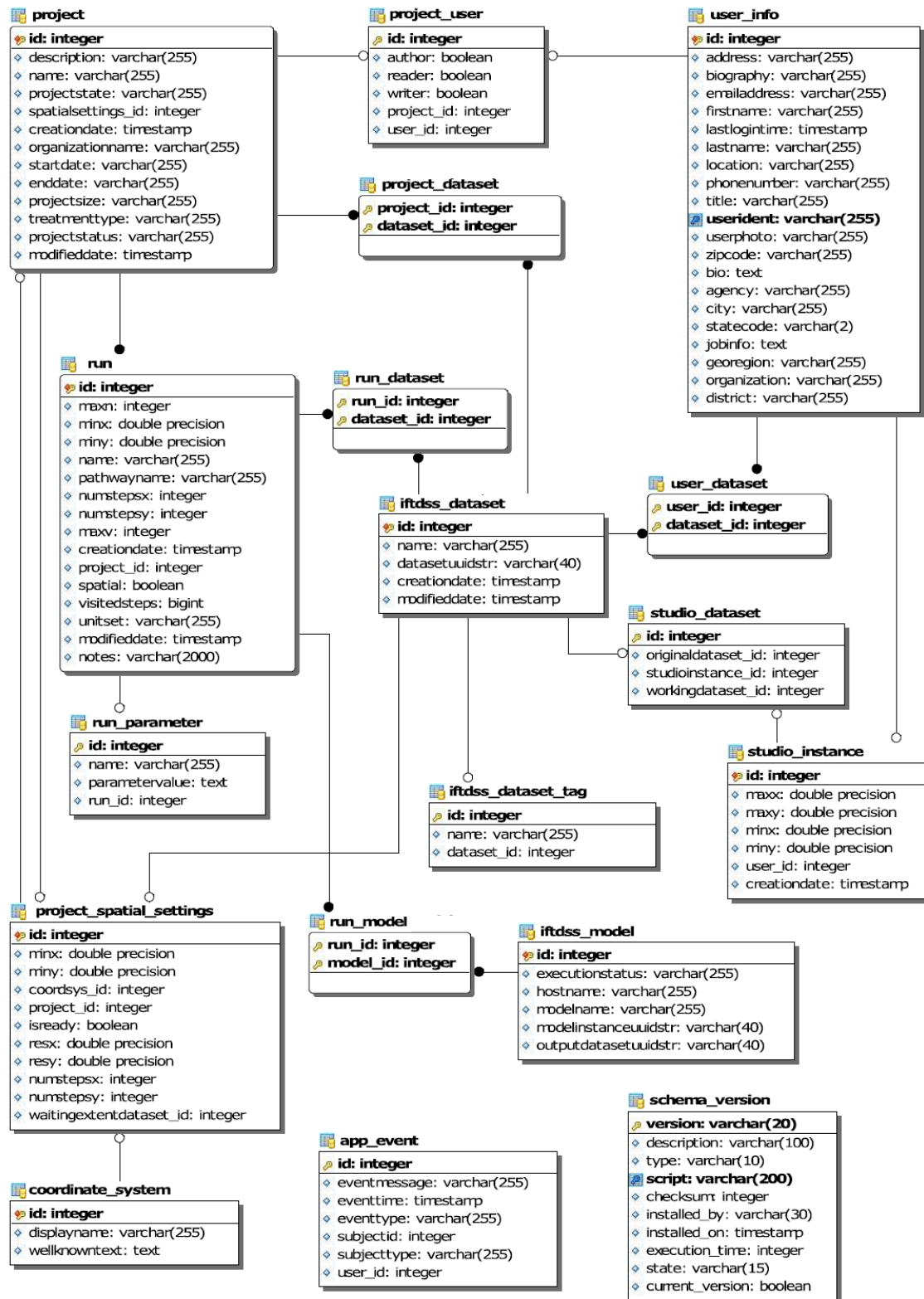


Figure 3-8. The IFTDSS conceptual model database schema.

3.2 The Scientific Modeling Framework

The IFTDSS software integration framework uses the SMF, an SOA, for managing and integrating scientific models and data. The SMF is service-oriented in that it is composed of separate units, or services, that are loosely coupled and communicate with each other over network protocols.

SMF software services encapsulate scientific models and their data, compartmentalizing them from each other and from end-user applications such as IFTDSS. SMF services and models operate through programmatic interfaces. The graphical user interface (GUI) is left entirely to the application and is only very loosely coupled with the SMF, allowing all user interaction with models to be mediated by the application through the mechanisms of the SMF. For example, the IFTDSS web application takes user input and uses it to drive SMF services, triggering and monitoring the execution of SMF models, and eventually presenting the model output data to the user in the form of data tables, graphs, and maps.

The SMF core consists of a core software design and library; a set of scalable service implementations for managing metadata, data, and models; and a suite of tools and support libraries for application and model development.

The SMF has five major service components:

- **Web UI Library**, which enables dynamic web-page creation;
- **Executive**, which is the registry for SMF service hosts and models, manages job processing, and manages model parameters;
- **Data Storage**, which manages and stores multi-dimensional scientific data;
- **Aquisitor**, which imports data from external sources; and
- **Model Hosts**, which manage the execution of models.

The SMF also includes a database containing parameter and model definitions.

3.2.1 WebUI Library

The SMF WebUI Library defines and controls the user experience in the IFTDSS Application and is a registry for locating SMF service hosts and models. Subcomponents in the WebUI Library include plug-ins, a GIS Data Visualization service, and a Workflow Controller.

- Plug-ins generate various views (table, spatial, and graph) of information for presentation. The Spatial View plug-in also mediates all use of the GIS Data Visualization component.
- The GIS Data Visualization component supports the display of spatial data and maps. These services are based on the Web Map Service (WMS), which is a standard protocol for serving geo-referenced map images (generated by a map server using data from a GIS database) over the Internet. This service is widely used and was developed by the Open Geospatial Consortium (OGC).

- The Workflow Controller maintains communications with the IFTDSS application, drives the plug-ins, and directs workflow actions to the Executive and Data Management components.

3.2.2 Executive

The Executive component manages the work of the SMF by coordinating the work defined in the workflow that was invoked in response to an action by the user. The Executive receives information defining which workflow should be executed. The Executive responds by analyzing the models and data needed and causes model services to begin running with appropriate peer-to-peer data connections.

3.2.3 Scientific Data Storage

The Data Storage component contains both spatial and non-spatial data. It stores multidimensional data in network common data format (netCDF) and other structured data, manages data set IDs, and tracks data set groups. It manages data from the LANDFIRE database; treelist and polygon stand layers, such as those used by the USDA Forest Service's Forest Vegetation Simulator (FVS) software; and other forms of multidimensional data. The Data Storage component also handles time series and parameter ensemble data sets.

3.2.4 Acquisitor

The Acquisitor component manages the processes of importing user-provided files, acquiring external data, and storing those data in the Scientific Data Storage component.

3.2.5 Model Hosts

Models are the scientific and computational components of the IFTDSS and thus constitute the heart of the system. All other system components were designed to support the operation of the models and the examination of the model outputs. Existing models can be accessed through the use of a model wrapper or through a Web Service adaptor. In addition, new models can be written to plug directly into the IFTDSS.

One goal of the IFTDSS project was to facilitate the process of adding new models to the system through the use of standardized interfaces and specifications. By publishing and supporting standardized interfaces, the IFTDSS project makes it possible to modify and easily integrate existing applications and services with the new system. In addition, new applications and services can be designed to support the standardized interfaces. These standardized interfaces include the following specifications:

- Java interface specifications. Modules written in Java can easily be integrated with these interfaces.
- Command-line application standard. Native applications that can run from a command line can support some standard command-line arguments and input and output formats.

- Web services. Standard web service method names and input and output types are specified. The IFTDSS system demonstrates this interface through RESTful⁷ web services.

In addition, the system can support specifications in XML syntax for all standard data types. Applications and services that support these standard data types and standard interfaces will be simple to integrate into the framework.

3.3 The Scientific Models

In the IFTDSS, a single model is defined as the source code, or model calculation software, that performs a specific mathematical algorithm; a module is a collection or grouping of models. However, the SMF doesn't differentiate between a single model and a module as long as the model or module accepts a well-defined set of input parameters and produces a well-defined set of output parameters. Each model also defines an "aperture," or size of its unit of work. Models may operate on a single coordinate at a time, all the coordinates in the given data set, or something in between. The SMF divides the input data as needed and performs the model calculations on each "clump" of data, running model executions in a loop or in multiple threads as needed.

Each model parameter defines a single element or layer of data in the SMF. Example parameters include wind speed, elevation, canopy height, and flame length. Variations of similar values—for example, "Flame Length," "Flame Length at Head," and "Flame Length at Back"—are treated as distinct parameters, although the SMF can convert between closely related parameters in some cases. Parameters are not fixed to a specific unit of measure; instead, whenever models and applications interact with data, the units of measure must be specified and the SMF automatically converts between units.

Data values for input or output parameters are stored as data sets, which are managed by the SMF Data Storage server. The Data Storage server provides a database optimized for storing scientific data, including spatial information.

The SMF system can have one or more Model Host servers, each of which provides an execution environment for the model calculations. Model Host servers are services within the SOA framework of the SMF, which communicate with other SMF components across a network. This way, Model Host servers can be located in the same machine as the rest of the application, or distributed across the network at other locations.

The SMF organizes models into model packages. A model package is a specially-formatted ZIP file containing everything necessary to run the model within the SMF, plus a special XML file describing the contents of the package. The XML file lists the input and output parameters that each model uses, as well as instructions for how the Model Host should communicate with the model calculation software. The SMF can only communicate natively

⁷ Representational state transfer (REST) describes architectures that use HTTP or similar protocols by constraining the interface to a set of well-known, standard operations (like GET, POST, PUT, and DELETE for HTTP). A RESTful web service is a web service implemented using HTTP and the principles of REST.

with model calculation software written in Java, JavaScript, or Python, though it also supports the use of wrappers written in one of those languages that can interface with any other program, such as a command-line batch version of a model. In the future, other languages and execution environments can be supported as well.

4. Software Implementation Approach

Working from the IFTDSS design, the IFTDSS Development Team implemented the IFTDSS software using a modified agile software development methodology. During the development process, several early releases were made available to a test user group to work with and provide feedback. This feedback was used to revise the requirements and design. This section describes both the agile software development and feedback processes used.

4.1 Agile Software Development Process

The Agile software development process uses a group of software development methods based on iterative and incremental development. In agile development, requirements and solutions evolve through collaboration between interdisciplinary teams. This method promotes adaptive planning, evolving development objectives, and a time-constrained iterative approach, and encourages rapid and flexible response to change. It also is a process in which a development team can self-organize and make changes quickly. The Agile manifesto⁸ states

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value

- *Individuals and interactions over processes and tools*
- *Working software over comprehensive documentation*
- *Customer collaboration over contract negotiation*
- *Responding to change over following a plan*

That is, while there is value in the items on the right, we value the items on the left more.

While there are many variations of formal agile methods, the method used in the development of IFTDSS is a modification of formal methods designed to facilitate a broader community of subject matter experts than is normally involved in software development projects. In this approach, development occurs in “sprints.” A sprint is a set period of time during which specific work has to be completed and made ready for review. A scrum master is the facilitator for the development team that uses the agile process (scrum is one form of the agile development method). The scrum master manages the process for exchanging information. Sprints consist of four activities.

1. Create product backlog
2. Sprint planning
3. Sprint development
4. Sprint review

Each of these activities is discussed below.

⁸ Manifesto for Agile Software Development. <http://agilemanifesto.org/>

4.1.1 Create Product Backlog

The initial step is to create Stories (small development tasks) based on requirements, the functional specification, user feedback, and anything new that may have come up from the previous sprint. These stories are written on Post-It notes and are sorted into functional area on the Product Backlog white board.

4.1.2 Sprint Planning

Each sprint begins with planning. For IFTDSS, the sprints are generally two weeks long. The sprint planning effort includes a sprint planning meeting and a developer level-of-effort meeting.

The sprint planning meeting is typically two to four hours in length and is attended by the product owner (project manager), lead developer, subject expert, task manager, quality assurance (QA) lead, and scrum master. During this meeting, attendees

- Decide how long the sprint should be (usually two weeks) and what the goal of the sprint is (for example, the user should be able to run the FlamMap model and view limited output).
- Based on the sprint goal and length, decide which stories should be worked on for the sprint (this is referred to as the sprint backlog).
- Make tentative development assignments.

The developer level-of-effort meeting is typically two to four hours in length. The developers who will be working on this sprint decide how long (a rough level of effort estimate) each story from the sprint backlog will take to develop. If the amount of work is more than the sprint length or is not enough, then everyone meets to discuss what should be added or dropped. Also during this meeting, final development assignments are made.

4.1.3 Sprint Development

During the sprint development activity, information exchange is facilitated through the use of scrums. Two types of scrums are used.

- Daily scrums. Each day, the developers and the scrum master meet for 15 minutes to answer three questions (non-developers are free to join, but their attendance is not required)
 - What was accomplished yesterday
 - What is going to be done today
 - What, if anything, is blocking them from finishing their tasks
- Scrum of scrums. If there is anything that needs to be discussed outside of those three questions, then the developers who need to discuss the topic meet after the scrum.

4.1.4 Sprint Review

At the end of each sprint, all team members review what was developed during the sprint. They also meet for a sprint retrospective, during which they

- Demonstrate the sprint deliverable
- Discuss the agile process – what worked and what needs improvement

After these meetings, team members start the next sprint.

4.1.5 Summary the IFTDSS Development Process

In summary, an agile methodology was used in developing the IFTDSS, which included these elements.

- Two- to three-week sprints (timeboxing)
- Regular deployments to alpha server and demos
- Half-day or all-day sprint planning meetings
- Follow-up requirements analysis meetings
- 15-minute morning standup meeting (“The Scrum”)
- Small, self-organized teams
- Lots of face-to-face discussion
- Planning board (a physical board, mirrored in JIRA⁹, a bug/issue database)
- Centralized software configuration management, branching as necessary (rarely)
- Continuous integration (Jenkins CI¹⁰)
- Variety of automated testing methods: unit, integration, in-browser
- QA after each task is completed
- Automated code quality analysis (Sonar¹¹)
- Multiple deployment environments with one-click redeployment
- Emphasis on always-working software
- Regular review of JIRA
- Willingness to redesign
- Regular refactoring

⁹ JIRA is a tracking system designed for development teams. It is used to track bugs and tasks, link issues to related source code, plan agile development, monitor activity, report on project status, and more. See <http://www.atlassian.com/software/jira>

¹⁰ Jenkins CI is an open-source continuous integration server that provides over 400 plug-ins to support building and testing software. See <http://jenkins-ci.org/>

¹¹ Sonar is the open-source platform for continuous inspection of code quality. See <http://www.sonarsource.com/products/>

4.2 Test User and Developer Feedback

Two groups of potential IFTDSS users were engaged in the development process: (1) those who would use IFTDSS for planning fuels treatments, and (2) those who develop the science and tools needed for fuels treatment planning. After each early release of the IFTDSS, individuals in these groups, who had agreed to participate as test users, were notified of the release and requested to provide comments. These requests were supplemented with workshops at meetings attended by those in the fuels treatment community and through webinars.

The IFTDSS feedback process is summarized in **Figure 4-1**. The process accommodates feedback from both the IFTDSS development team (internal) and test users (external). The JIRA issue-tracking and change-control system is an integral component of the development process and is used to manage reported bugs and feature enhancement requests. With all releases of IFTDSS, users can click a link on any page to expose a web form for submitting feedback directly into JIRA. Feedback received by telephone or email is also entered into JIRA to begin the resolution process.

Using the feedback received through this process, the IFTDSS Team can make midcourse corrections to the system design, subject to approval by the Contracting Officer's Technical Representative (COTR). This process ensures the usability, effectiveness, and efficiency of the software for those who will ultimately use it.

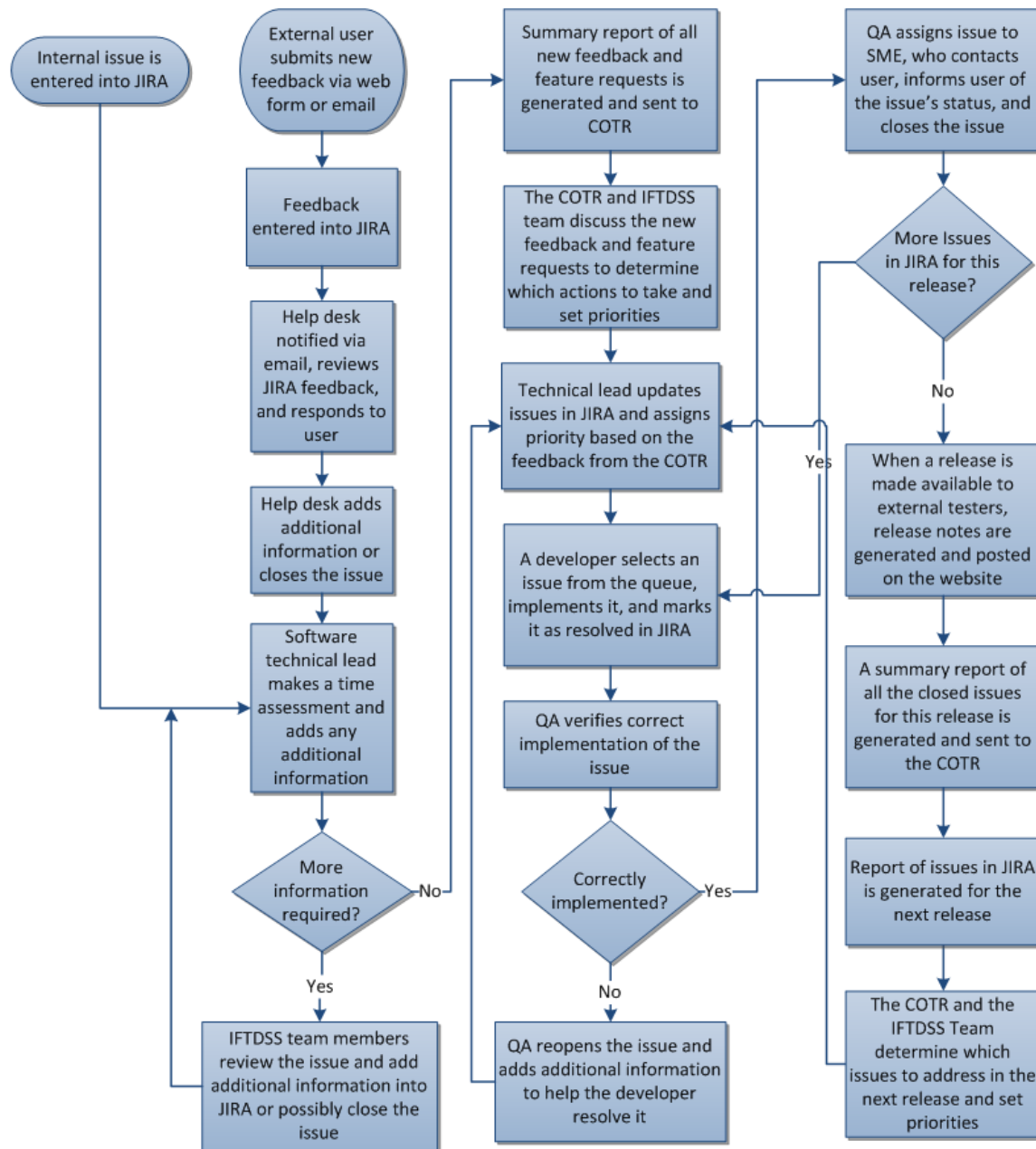


Figure 4-1. The IFTDSS feedback process.

5. IFTDSS Functional Features

The modeling tools available in IFTDSS are grouped and are accessible in three ways:

1. by IFTDSS workflow,
2. by model developer, and
3. by individual models available in IFTDSS.

The IFTDSS workflows provide sets of tools for fuels treatment, prescribed burn planning, assessing fire hazard, and assessing potential risk from fire. The modeling tools are also grouped by model developer; that is, the tools are organized by the science teams that developed the models, and includes the model type and the outputs produced. The third way to access models within IFTDSS is to view an alphabetical listing of all models.

Access to these models is initiated by creating a new project and selecting from the three groups shown in **Figure 5-1**.

The screenshot shows a web interface for creating a new project. At the top, a green notification bar says "Created project 'IFTDSS Example'". Below this, the heading "Choose the type of run you would like to create:" is followed by a "Start ▶" button and a "Back" button with a left arrow. Three folder icons represent the selection options: "By IFTDSS Workflows", "Fire and Fuels Application (FFA)", and "All Available Modules in IFTDSS". A text box on the right provides a detailed description of these three access methods.

Choose the type of run you would like to create:

Start ▶ Back

By IFTDSS Workflows

Fire and Fuels Application (FFA)

All Available Modules in IFTDSS

The modeling tools available in IFTDSS are grouped in three ways: 1) by IFTDSS workflow, 2) by tools developed by the FERA Team, and 3) by all modules available in IFTDSS. The IFTDSS workflows provide access to the tools in IFTDSS organized by specific fuels planning tasks (i.e., prescribed burn planning, assessing fire hazard, assessing potential risk from fire, and assessing fuels treatment strategies). The tools developed by the FERA Team provides access to only the modeling tools developed by the FERA Team, organized by the types of outputs each tool produces. The directory containing all modules within IFTDSS provides access to all of the modules available organized by the module type.

Figure 5-1. Selecting how models will be accessed in a new project.

Descriptions of the workflows and tools available through these three access methods are provided in the following subsections.

5.1 IFTDSS Workflows

Leading up to the development of IFTDSS, efforts were made to understand the decision support needs and workflow processes involved in fuels treatment planning and management. As a result of these efforts, the following four workflows were identified and have been implemented in IFTDSS Version 2.0:

1. The **Hazard Analysis Workflow** is used to identify potentially hazardous areas across a landscape. The focus of this workflow is to identify areas across a landscape where fuels treatment analysis may be warranted based on potential fire hazard. IFTDSS provides tools that support this workflow.
2. The **Risk Assessment Workflow** provides a first-approximation probabilistic risk assessment for fuels treatment planning.
3. The **Fuels Treatment Workflow** (a) simulates fuels treatment placement in areas of high fire hazard within an area of interest, (b) simulates post-treatment influences on fire behavior and fire effects potentials, and (c) evaluates the temporal durability of fuels treatments, that is, how long, in years to decades, a treatment will continue to reduce adverse fire behavior and fire effects within an area of interest.
4. The **Prescribed Burn Planning Workflow** provides the information needed to plan and document a proposed prescribed fire. IFTDSS provides tools that support this workflow; with these tools, users can
 - calculate the probability of ignition from lightning or a firebrand
 - assess and calculate fire behavior
 - assess and plan fire containment
 - calculate fire effects
 - create a prescribed burn plan (including printing out a Word document with many elements filled in by IFTDSS)

The following subsections provide an overview of each of these workflows as implemented in IFTDSS 2.0 and the data acquisition and preparation function integrated into each workflow.

5.1.1 Hazard Analysis Workflow

The hazard analysis workflow provides tools for performing a current-condition assessment of fire hazard within an area of interest. The objective of this workflow is to spatially identify high fire hazard locations within an area of interest. High fire hazard is expressed by high potential fire behavior (e.g., flame length, rate of spread, and fireline intensity) and/or undesirable fire effects (e.g., tree mortality and emissions).

Identifying high fire hazard and prioritizing potential fuels treatment areas based on fire hazard is an important task for fuels treatment planners. Thus, fire hazard analysis can be viewed as an initial step in the fuels treatment and/or prescribed burn planning process(es), and

can be performed across many geographic scales (e.g., national, district, watershed). Once high fire hazard has been identified, the user would then use additional tools within the IFTDSS to conduct further analyses, such as identifying values at risk within the landscape, or determining where to place fuel treatments to mitigate high fire hazard.

The term “fire hazard” is defined as an act or phenomenon with the potential to do harm (National Research Council, 1989; Keane et al., 2010). Fire hazard in this context is expressed as potential fire behavior (such as flame length), which is related to fuel properties within the area of interest (Keane et al., 2010). Fire hazard is often expressed independently of weather (Hardy, 2005), yet the potential effects of fire on values at risk are influenced by weather and its relationship to fuel moisture.

IFTDSS provides four modules available for use in the hazard analysis workflow.

1. The IFT-FlamMap module computes potential fire behavior potentials across a user-defined landscape using a constant weather scenario and spatial landscape data from LANDFIRE data sets. IFT-FlamMap uses the FlamMap 3.0 (Finney, 2006) algorithms to simulate potential head fire behavior characteristics such as flame length, rate of spread, and fire line intensity in a spatial context.
2. The IFT-MTT module simulates fire growth building on the functionality found in IFT-FlamMap. In this module, fire growth is simulated using the MTT algorithms and LANDFIRE data sets to provide spatial landscape data and a constant weather scenario.
3. The IFT-RANDIG module simulates burn probability potentials across a user-defined landscape. IFT-RANDIG uses the MTT algorithms from the IFT-MTT module run numerous times across the landscape of interest (Finney, 2002). Burn probability is produced by simulating a user defined number of randomly located ignitions within the area of interest and recording the number of pixels that burn for each ignition. The probability a pixel is burned given a random ignition within the landscape is calculated by dividing the number of times an individual pixel burns by the number of random ignitions. In this module, burn probability maps are created that represent the composite burn probability (burn probabilities for all flame lengths) and the probability that a pixel will burn with a specified flame length.
4. The IFT-Consume landscape module simulates fire effects (fuel consumption, smoke emissions, heat release) across an area of interest. Spatial outputs such as fuel consumption are computed for each pixel using a single fuel moisture scenario and the Fuel Characteristics Classification System (FCCS) Fuelbed map from LANDFIRE as data inputs.

Each of these four modules produces digital maps that represent the current fuels on the ground and how these fuels might burn under specified wind and pixel-specific environmental constraints. These digital maps provide useful decision support on where potential fire behavior and fire effects. **Figure 5-2** provides an overview of the functionality in the Hazard Analysis Workflow.

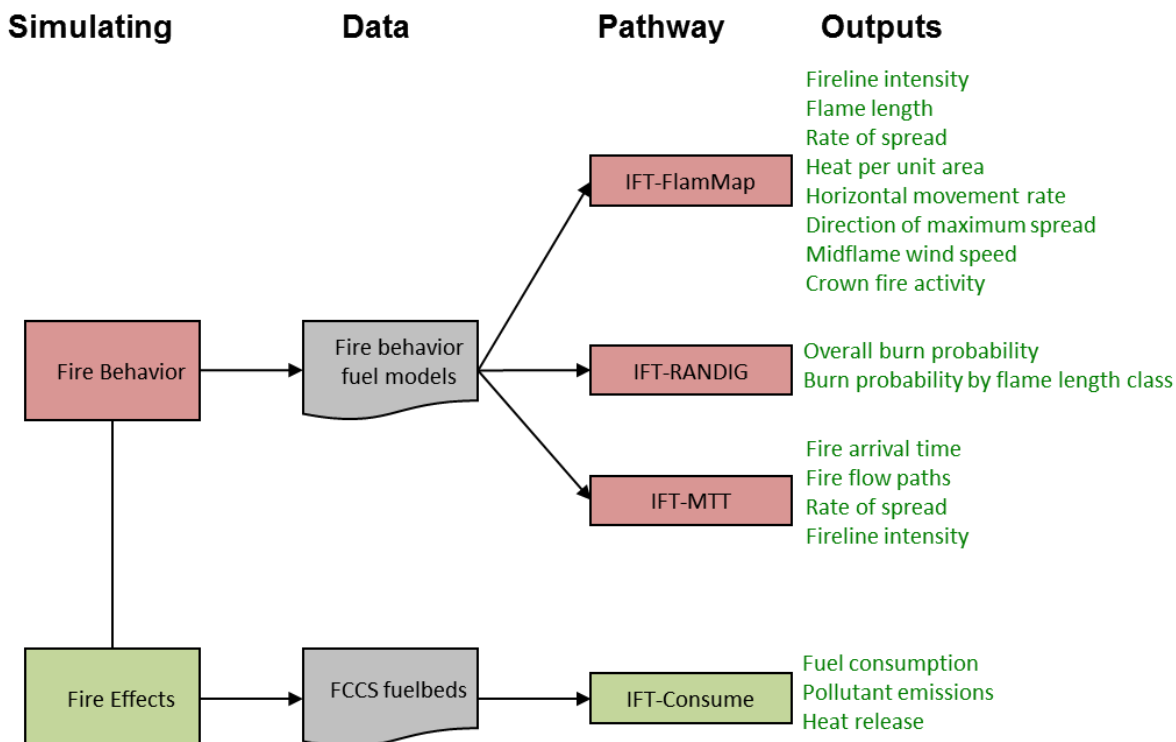


Figure 5-2. Overview of the hazard analysis workflow.

5.1.2 Risk Assessment Workflow

There is no universally accepted framework for assessing the social, economic, and ecological risks resulting from fire in the landscape. However, recent efforts have been made to develop a risk assessment process that can be used to provide information useful for prioritizing where fuels treatments and mitigation measures may be warranted to address fire hazard and risk. IFTDSS provides two approaches for assessing fire hazard and risk across the landscape based on the methods described in the RMRS-GTR-235 *Wildfire Risk and Hazard: Procedures for the First Approximation* (Calkin et al., 2010).

The processes proposed by Calkin et al. (2010) and modified for use in IFTDSS are designed to develop a strategic-level, first approximation of how fire likelihood and fire behavior potentials across landscapes influence risk to social, economic, and ecological values within an area of interest. The Calkin et al. (2010) approach provides a quantitative risk framework that approximates the expected loss and/or potential ecological benefits to valued resources (values at risk) from wildfire. In this process, burn probabilities and fire behavior potentials (described in Section 5.1.1) are estimated using fire simulation modules. The modeled output is coupled with data on human and ecological values at risk using fire-effects response functions to estimate the expected loss or potential benefit resulting from fire.

We choose the Calkin et al. (2010) approach that incorporates risk assessment research by Mark Finney and Alan Ager for our first set of risk assessment modules for the following reasons:

- First, the Calkin et al. (2010) approach uses the best developed quantitative procedures for assigning risk across landscapes in a fire management context.
- Next, these procedures use tools and functionality currently implemented in the IFTDSS.
- Finally, we felt that the Calkin et al. (2010) approach provides easy-to-understand quantitative measures of the hazards of burning, while also including quantitative metrics for evaluating potential ecological benefits of fire on the landscape.

Additional risk assessment strategies will be implemented as IFTDSS development continues.

IFTDSS provides two approaches for assessing risk: risk assessment by worst-case flame length and risk assessment by burn probability.

Risk Assessment by Worst-Case Flame Length

In this module, risk is defined as the expected net value change within an area calculated as the product of (a) the probability that the area represented by the pixel will burn given a random ignition within the project area, and (b) the resulting change in financial or ecological value (response function) if the area represented by the pixel burns with a specific flame length.

This method uses the response functions developed by Calkin et al. (2010), modeled flame lengths from the FlamMap fire behavior module, and burn probabilities from the IFT-RANDIG burn probability simulator to estimate the likelihood of the area represented by the pixel burning, and the potential consequences if the area represented by the pixel is burned by a head fire.

This approach is referred to as the “worst case” estimation of fire risk because it is based on a single IFT-FlamMap run, where the areas represented by every pixel are all always assumed to burn under the worst case (i.e., by a head fire). However, for this approach, the IFT-FlamMap-RANDIG burn probability simulator provides information as to whether the area represented by a pixel will burn regardless of flame length; that is, the area represented by a pixel can be burned as a backing fire, flanking fire, or head fire in the IFT-FlamMap-RANDIG simulations. This approach may overestimate the degree of damage to the value at risk in an area represented by an individual pixel.

Risk Assessment by Flame Length Probabilities

In this module, risk is defined as the expected net value change within an area calculated as the product of (a) the probabilities that the area represented by the pixel will burn (using user-defined flame lengths and flame length classes—low, medium, high, and very high) given a random ignition within the project area, and (b) the resulting change in financial or

ecological value (response function) if the area represented by the pixel burns for each user-defined flame length class.

This method uses the response functions developed by Calkin et. al. (2010) and modeled flame length burn probabilities from the IFT-FlamMap-RANDIG burn probability simulator to estimate the likelihood of the area represented by the pixel burning, and the potential consequences if the area represented by the pixel is burned by a backing fire, a flanking fire, or a head fire.

This approach differs from the worst-case flame length approach in that it considers the likelihood of a fire burning as a backing fire, a flanking fire, or a head fire given a random ignition in the landscape when determining the potential losses or benefits for an area represented by a pixel burning.

Products of the Risk Assessment Modules

The products from each of the risk assessment models are a series of digital maps that provide information regarding where the fire is likely to burn given a random ignition, the potential hazard if the area burns, and the potential losses and or benefits to values within the landscape if the area burns. The results from the risk assessment tools in IFTDSS can provide information useful for evaluating and prioritizing where to place fuel treatments to reduce fire hazard and risk to valued resources. **Figure 5-3** provides a brief overview of the data inputs and the products produced by the risk assessment workflow.

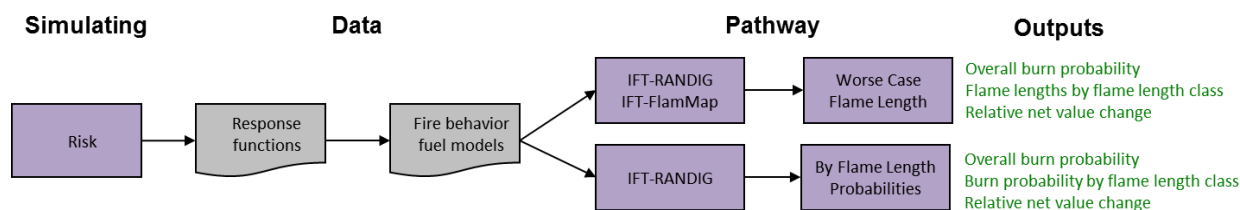


Figure 5-3. Overview of the risk assessment workflow.

5.1.3 Fuels Treatment Workflow

Fuels treatments are designed to lower hazardous fire behavior potentials and restore ecosystem resiliency temporally and spatially. Currently, large areas in the United States are considered to be in a hazardous fire condition due to longer, hotter fire seasons; dense, contiguous forest canopies; and high levels of surface downed and dead woody fuels (U.S. Congress, 1998). Due to hotter conditions and more fuel, these areas are burning hotter and more intensely. As a result, fire behavior that was once considered abnormally high is now common (Brown, 1985; Mutch, 1994; Ferry et al., 1995). To mitigate the effects of abnormally high fire behavior in these landscapes, land managers increasingly turn to fuels treatments, including prescribed burning, to reduce fuel quantities. Land managers cannot control climate, but they can manipulate fuel levels to lower the amount of fuel available to burn.

The goals of the fuels treatment workflow In the IFTDSS are to

1. provide tools and processes for fuels treatment planners to evaluate potential fuels treatment options including no treatment, pile and burning, thin from below, thin by diameter at breast height (Dbh), pile burn surface fuel, and simulate a broadcast prescribed burn;
2. identify where fuels treatments may have the greatest influence for mitigating wildland fire at the stand and landscape scale; and
3. investigate the potential effectiveness of fuels treatments across spatial and temporal scales.

Fuels Treatment for Individual Stands Pathway

The stand level, or point, fuels treatment pathway in IFTDSS contains fuels treatment algorithms from the Fire and Fuels Extension to the Forest Vegetation Simulator (FVS-FFE; Reinhardt and Crookston, 2003). Within the IFTDSS, the FVS-FFE is used to simulate potential changes in fuel quantity and potential fire behavior due to treating or manipulating the vegetation. At the stand level (single point), IFTDSS users can simulate vegetation changes due to:

- Mastication – a mechanical fuels treatment where fuels such as small trees, brush and other vegetation are chopped, ground, or chipped into small particles that are then spread across the ground to make them less flammable and reduce subsequent fire behavior potentials when burned, such as increased flame length.
- Thin from below – a mechanical treatment where all small trees and shrubs below a designated size limit are removed
- Thin a species across a Dbh range – a treatment where trees for a particular species or all species present within an area are thinned to a specified percentage canopy cover target.
- Pile burn surface fuel – in this treatment, fuel is concentrated into piles and burned within the stand, in this treatment the assumption is made that 80 percent of the fuel from 70 percent of the stand is concentrated into piles covering ten percent of the area (Reinhardt and Crookston, 2003).
- Thin with fuel piled and burned – in this treatment option, the trees are first thinned to specified conditions, then piled and burned.
- Prescribed burn – in this treatment, the FVS-FFE module simulates how a broadcast burn would influence fuel levels across the landscape

With the point-based Forest Vegetation Simulator (FVS) tools, users can identify which treatment may provide the best option for reducing fuels within a single stand. This option also allows users to investigate how long a specific treatment will last. That is, algorithms in the FVS-FFE can simulate potential vegetation change due to a specified treatment, and also simulate vegetation growth over time after the treatment. Using the point-based fuels treatment options, users can identify fuel treatments to mitigate, or lower, potential fire behavior and to

investigate how long those treatments might continue to lower potential fire behavior as vegetation grows after treatment but do not evaluate how fuels treatment influences in a spatial context.

Fuels Treatment Across a Landscape Pathway

Using the fuels treatment across a landscape pathway, users can spatially assess where to locate fuels treatments and evaluate how effective a fuels treatment is at mitigating fire behavior potentials in a spatial context. In IFTDSS Version 2.0, users can simulate fuels treatments in a landscape by manually drawing polygons (**Figure 5-4**). Treatments are simulated by manually editing a LANDFIRE .lcp file using a set of editing tools. With these editing tools, users can enter values (fuel model, canopy height, canopy base height, canopy bulk density, canopy coverage) within the fuels treatment polygon; the module then alters the polygon to reflect a fuels treatment. Spatial fire behavior modules such as IFT-FlamMap and IFT-MTT are run across the untreated and the treated landscape to compare possible effects of treating the fuels on subsequent fire behavior. IFTDSS produces digital maps of difference and percentage difference that identify where change has occurred with the landscape.

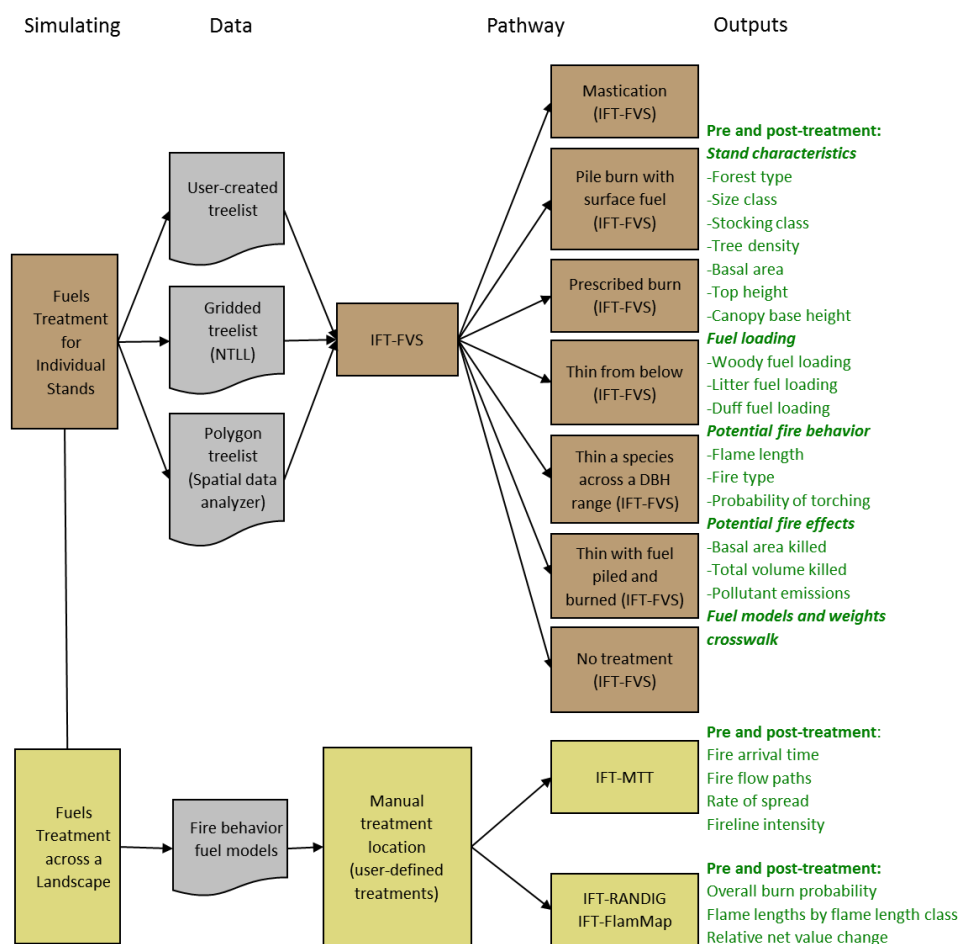


Figure 5-4. Overview of the fuels treatment workflow.

5.1.4 Prescribed Burn Planning Workflow

Prescribed burns are planned to meet management and operational objectives in accordance with the *Interagency Prescribed Fire Planning and Implementation Procedures Guide* (U.S. Department of Agriculture and U.S. Department of the Interior, 2008). All prescribed fires require an approved plan that must be followed when a burn is conducted. The prescribed fire burn plan is the legal document that provides an agency administrator with the information needed to approve a prescribed fire. The size and complexity of a prescribed fire project determine the level of effort and detail to be included in the plan; however, each plan must specifically address 21 standard elements (i.e., required information) in the prescribed fire template (which is provided in the guide and also with IFTDSS).

Prescribed burn planning requires a burn planner to collect data, run fire behavior and fire effects simulations over a range of environmental variables, and make decisions that enable the burn plan objectives to be met during the process of maintaining control of the fire. To complete these tasks, prescribed burn planners typically use a variety of software tools with various data requirements. In IFTDSS, model use and data structures have been simplified to streamline the prescribed burn planning process. Modeled output of fire behavior and fire effects is provided in a concise, user-friendly format that can be easily exported to Microsoft Word and Excel.

IFTDSS Modules that Support Prescribed Burn Planning

IFTDSS provides several point fire behavior modules that are useful for prescribed burn planning. These modules include the following:

1. IFT-surface uses the algorithms and recreates the common fire behavior attributes (such as rate of spread and flame length) found in the SURFACE module of the standalone fire behavior prediction system BehavePlus (Heinsch and Andrews, 2010).
2. The IFT-crown is another aspatial fire behavior module that uses BehavePlus algorithms to simulate potential crown fire behavior on a single point within a landscape.
3. The IFT-surface+size module links the algorithms used in the BehavePlus SURFACE and SIZE modules to link fire behavior and fire size simulations.
4. The IFT-FlamMap point module uses the algorithms in the FlamMap desktop software to model fire behavior at the individual stand level. The IFT-FlamMap point module differs from IFT-surface in that IFT-FlamMap point links the algorithms for modeling surface fire behavior to the algorithms for modeling crown fire behavior through the use of a surface to crown fire transition algorithm (Scott and Reinhardt, 2001).
5. The IFT-FCCS surface fire behavior module provides an alternative methodology for simulating potential fire behavior characteristics using a mathematical reworking of Rothermel's original fire spread model (Sandberg et al., 2007). The IFT-FCCS surface fire behavior module uses FCCS fuelbed information (Ottmar et al., 2007) coupled with the reformulated version of the Rothermel fire spread model to estimate fire behavior characteristics such as flame length, rate of spread, and fireline intensity.

6. IFT-FlamMap is the spatial fire behavior prediction module used for prescribed burn planning in Version 2.0. The IFT-FlamMap module computes potential fire behavior potentials across a user-defined landscape using a constant weather scenario and spatial landscape data from LANDFIRE data sets. IFT-FlamMap uses the FlamMap 3.0 algorithms (Finney, 2006) to simulate potential head fire behavior characteristics such as flame length, rate of spread, and fireline intensity in a spatial context.

IFTDSS simulates potential fire effects using IFT-FOFEM and IFT-Consume to predict fuel consumption, smoke emissions, and tree mortality.

The IFT-FOFEM module provides tools to simulate potential fuel consumption, smoke production, and tree mortality caused by prescribed fire or wildfire. In order to calculate consumption and emissions, cover type, fuel loading, and moisture information are needed. The output variables include amount of fuel consumed during fire, post-burn fuel loading, emissions released during flaming and smoldering combustion, and total flaming and smoldering time. In order to calculate tree mortality, tree species, stand characteristics, and fire behavior information are needed. The output variables include percentage mortality, stand basal area pre- and post-fire, and stand canopy cover pre-and post-fire.

IFT-Consume (Ottmar et al., 1993; Prichard et al., 2006) is a decision-making tool that assists planning for prescribed burns and wildfires using realistic fuels data. IFT-Consume predicts fuel consumption, pollutant emissions, and heat release based on input fuel characteristics, lighting patterns, fuel moistures and other environmental variables.

The Prescribed Burn Planning Workflow

Figure 5-5 provides an overview of the prescribed burn workflow. The modules in these workflows can be used to obtain the information needed to address several of the prescribed burn plan elements, including

- Element 3 (complexity analysis)
- Element 4 (description of burn area)
- Element 5 (burn objectives)
- Element 7 (burn plan prescription)
- Element 15 (ignition plan)
- Element 16 (holding plan)
- Element 17 (contingency plan)
- Element 19 (smoke management and air quality restrictions)
- Appendix A (maps; vicinity and project)

The matrix in **Table 5-1** lists the 21 standard prescribed burn plan elements and the tools available in IFTDSS to help complete each element.

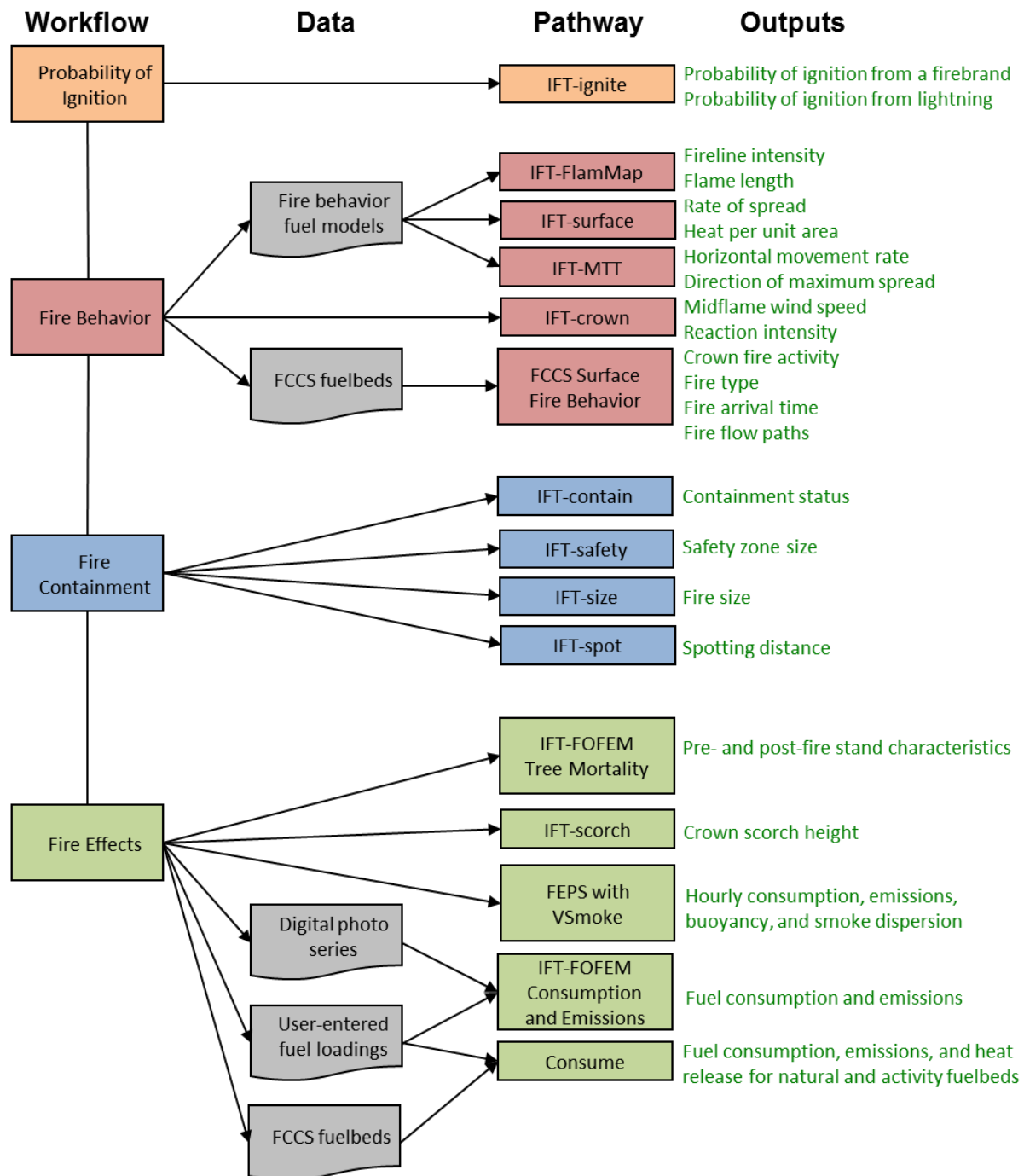


Figure 5-5. Overview diagram of the prescribed burn workflow – IFTDSS Version 2.0.

Table 5-1. Standard prescribed burn plan elements and the tools available in IFTDSS to help complete each element.

*IFT-BehavePlus *FCCS †IFT-FlamMap †IFT-FOFEM *Consume †IFT-FireFamilyPlus ○ = facilitate in decision making ● = outputs needed for burn plan	Fire Behavior					Fire Effects				Fire Containment			Probability of Ignition		Historical Fire Weather	Data and Mapping Tools		
	Surface fire behavior ^a	Surface fire behavior for FCCS fuelbeds ^b	Crown fire behavior ^a	Fire behavior for individual stands ^c	Fire behavior across a landscape ^c	Consumption and Emissions ^d	Tree Mortality ^d	Crown scorch height ^a	Natural fuels consumption ^e	Spotting distance ^a	Containment resources ^a	Safety zone size ^a	Fire size and spread ^a	Probability of ignition from a firebrand ^a	Probability of ignition from lightning ^a	Fire weather statistics ^f	Data Studio (project area of interest maps)	LANDFIRE Data (Fuel Model & Topography)
Element 1: Signature Page																		
Element 2: Agency Go/No-Go Checklist																		
Element 3: Complexity Analysis Summary	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		
Element 4: Description of Prescribed Fire Area					●												●	●
Element 5: Objectives	○	○	○	○	○	○	○	○	○									
Element 6: Funding																		
Element 7: Prescription	●	●	○	●	●	○	○	●	○	●				●		●		
Element 8: Scheduling																		
Element 9: Pre-Burn Consideration and Weather																○		
Element 10: Briefing Checklist																		
Element 11: Organization and Equipment																		
Element 12: Communication																		
Element 13: Public and Personnel Safety and Medical																		
Element 14: Test Fire																		
Element 15: Ignition Plan	○																	
Element 16: Holding Plan	○	○	○	○	○					○	○		○	○			○	
Element 17: Contingency Plan	○	○	○	○	○	○		○		○	○	○	○	○		○		
Element 18: Wildfire Conversion																		
Element 19: Smoke Management and Air Quality						●		●								○		
Element 20: Monitoring																		
Element 21: Post-Burn Activities																		
Appendices: Appendix A. Maps (Vicinity and Project)					●													

5.1.5 Data Acquisition and Preparation

Regardless of workflow or the scale of analysis, fuels treatment specialists require vegetation data of high quality to support fuels treatment planning. In addition, the appropriate data required to meet the analysis objectives must be identified prior to conducting an analysis. For many analyses, geophysical (elevation, slope, aspect) and weather data may also be required. During Phase II of the STS Study, the issues involved with obtaining and preparing vegetation data for fuels treatment planning were identified and documented (Rauscher, 2008).

IFTDSS provides users with the following data sources: treelist data (FSVeg); the LANDFIRE data products; user-supplied data in treelist or LANDFIRE formats; and user-supplied data for stand-level analysis.

5.2 Developer-Designed Functionality

IFTDSS supports the organization of tools by developer design workflows. These workflows may be a single calculation or a series of calculation implemented in the developer's original tool set or application. So far, one developer tools set has been incorporated in

IFTDSS: the Fire and Environmental Research Applications (FERA) Team's Fire and Fuels Application (FFA) tools. FERA is a USDA Forest Service research team focusing on fuels and fire and landscape ecology. IFTDSS supports several FERA tools:

- **Consume.** Predicts fuel consumption, pollutant emissions, and heat release based on a number of factors, including fuel loadings, fuel moisture, and other environmental factors.
- **Fuel Characteristic Classification System (FCCS) and FCCS fuelbeds.** Stores and classifies fuels data as fuelbeds, calculates physical characteristics of fuels based on fuelbed data, and calculates fire potentials based on the intrinsic properties of fuels.
- **Fire Emission Production Simulator (FEPS).** Predicts fuel consumption, emission rates, and heat release characteristics of prescribed burns and wildland fires. Total burn consumption values are distributed over the life of the burn to generate hourly emission and release information.
- **Digital Photo Series (DPS).** Provides the Natural Fuels Photo Series data electronically. Users can link to the DPS from within IFTDSS to obtain fuel loading information for a selected situation to replace default values.

5.3 Direct Access

To assist users that wish to perform calculations with a single tool instead of going through the entire workflow process, IFTDSS provides direct access to a range of tools through standalone user interfaces. These tools are accessible through the run type selection screen shown in Figure 5-1. **Table 5-2** lists the 71 individual calculations that can be performed in IFTDSS Version 2.0. **Table 5-3** provides a summary description of the original models the IFTDSS calculations are based on.

Table 5-2. Separate calculations that can be performed in IFTDSS.

Page 1 of 2

Calculation (Module)	
1.	All fuel characteristics (FCCS)
2.	All potentials (FCCS)
3.	Available fuel potential (FCCS)
4.	Calculate burn probability across a landscape (IFT-RANDIG)
5.	Calculate consumption and emissions (IFT-FOFEM)
6.	Calculate crown fire behavior (IFT-crown)
7.	Calculate fire behavior across a landscape (IFT-FlamMap)
8.	Calculate fire behavior for individual stands (IFT-FlamMap)
9.	Calculate fire effects across a landscape (IFT-Consume)
10.	Calculate minimum travel time (IFT-MTT)
11.	Calculate probability of ignition from a firebrand (IFT-ignite)
12.	Calculate probability of ignition from lightning (IFT-ignite)
13.	Calculate safety zone size (IFT-safety)
14.	Calculate spotting distance from a burning pile (IFT-spot)
15.	Calculate spotting distance from a wind-driven surface fire (IFT-spot)
16.	Calculate spotting distance from torching trees (IFT-spot)
17.	Calculate surface fire behavior (FCCS)
18.	Calculate surface fire behavior (IFT-surface)
19.	Calculate tree mortality (IFT-FOFEM)
20.	Computer generated treatment location (OptFuels – W. Lake Tahoe)
21.	Computer generated treatment location (OptFuels)
22.	Consume (activity fuelbeds)
23.	Consume (manual loadings, activity fuelbeds)
24.	Consume (manual loadings, natural fuelbeds)
25.	Consume (natural fuelbeds)
26.	Consume/FEPS (one activity fuelbed)
27.	Consume/FEPS (one natural fuelbed)
28.	Create a burn plan document
29.	Crown fire potential (FCCS)
30.	Estimate containment resources (IFT-contain)
31.	FCCS
32.	FCCS/Consume (activity fuelbeds)
33.	FCCS/Consume (activity fuelbeds) – batch
34.	FCCS/Consume (natural fuelbeds)
35.	FCCS/Consume (natural fuelbeds) – batch
36.	FCCS/Consume/FEPS (one activity fuelbed)
37.	FCCS/Consume/FEPS (one natural fuelbed)
38.	FEPS (manual)
39.	FEPS (pile burning)
40.	Fire Weather Statistics (IFT-FireFamilyPlus)

Table 5-2. Separate calculations that can be performed in IFTDSS.

Page 2 of 2

Calculation (Module)	
41.	Fuel consumption (activity fuelbeds, Consume)
42.	Fuel consumption (natural fuelbeds, Consume)
43.	Fuel loading (FCCS)
44.	Heat release (activity fuelbeds, Consume)
45.	Heat release (natural fuelbeds, Consume)
46.	Manual treatment location (user-defined treatments) (IFT-FlamMap)
47.	Manual treatment location (user-defined treatments) (IFT-MTT)
48.	Manual treatment location (user-defined treatments) (IFT-RANDIG)
49.	Manual treatment location (user-defined treatments) (Worst Case FL – Risk)
50.	Manual treatment location (FVS treatments)
51.	Mastication (IFT-FVS)
52.	NTLL to LCP (IFT-FVS)
53.	No Treatment (IFT-FVS)
54.	Pile burn surface fuel (IFT-FVS)
55.	Pollutant emissions (activity fuelbeds, Consume)
56.	Pollutant emissions (natural fuelbeds, Consume)
57.	Predict crown scorch height (IFT-scorch)
58.	Predict fire size and spread distance (IFT-size)
59.	Predict surface fire behavior, size, and spread distance (IFT-surface+size)
60.	Prescribed Burn (IFT-FVS)
61.	Risk Assessment - Worst Case Flame Length
62.	Risk Assessment - by Flame Length Probabilities
63.	SVS for No Treatment (IFT-SVS)
64.	Surface fire behavior (multiple fuelbeds, single scenario, FCCS)
65.	Surface fire behavior and potentials (FCCS)
66.	Surface fire potential (FCCS)
67.	Thin From Below (IFT-FVS)
68.	Thin a species across a Dbh range (IFT-FVS)
69.	Thin with fuel piled and burned (IFT-FVS)
70.	Total carbon (FCCS)
71.	Smoke dispersion (VSmoke on the BlueSky Cloud)

Table 5-3. Descriptions of models that IFTDSS calculations are based on.

Page 1 of 2

Model	Description
BehavePlus	The BehavePlus fire modeling system is a collection of models that describe fire behavior, fire effects, and the fire environment.
Consume	Consume 3.0 is designed to import data directly from the Fuel Characteristic Classification System (FCCS), and the output is formatted to feed other models and provide usable outputs for burn plan preparation and smoke management requirements. Additionally, training and a user's manual are available. Consume can be used for most forest, shrub, and grasslands in North America. http://www.fs.fed.us/pnw/fera/research/smoke/consume/index.shtml

Table 5-3. Descriptions of models that IFTDSS calculations are based on.

Page 2 of 2

Model	Description
FCCS	The Fuel Characteristic Classification System (FCCS) offers consistently organized fuels data along with numerical inputs to fire behavior, fire effects, and dynamic vegetation models. http://www.fs.fed.us/pnw/fera/fccs/index.shtml
FEPS	The Fire Emission Production Simulator (FEPS) manages data concerning consumption, emissions, and heat release characteristics of prescribed burns and wildland fires. http://www.fs.fed.us/pnw/fera/feps/index.shtml
FireFamilyPlus	FireFamilyPlus analyzes and summarizes an integrated database of fire weather and fire occurrence. It combines the functionality of the programs PCFIRDAT, PCSEASON, FIRES, and CLIMATOLOGY. FFP can be used to calculate fire danger rating indices and components and to summarize both fire and weather data. It offers options for jointly analyzing fire and weather data. http://www.firemodels.org/index.php/firefamilyplus-introduction/firefamilyplus-overview
FlamMap	FlamMap is a fire behavior mapping and analysis program that computes potential fire behavior characteristics (such as spread rate, flame length, and fireline intensity) over an entire landscape for constant weather and fuel moisture conditions. http://www.firemodels.org/index.php/national-systems/flammap
FOFEM	FOFEM (a First Order Fire Effects Model) is a computer program for predicting tree mortality, fuel consumption, smoke production, and soil heating caused by prescribed fire or wildfire. FOFEM provides quantitative fire effects information for tree mortality, fuel consumption, mineral soil exposure, smoke, and soil heating. http://www.firelab.org/science-applications/fire-fuel/111-fofem
FVS	The Forest Vegetation Simulator (FVS) is a family of forest growth simulation models. FVS answers questions about how forest vegetation will change in response to natural succession, disturbances, and proposed management actions. http://www.fs.fed.us/fmcs/fvs/
MTT	FlamMap's Minimum Travel Time (MTT) is a two-dimensional fire growth model that calculates fire growth and behavior by searching for the set of pathways with minimum spread times from a point, line, or polygon ignition source, keeping environmental (fuel moistures and winds) conditions constant for the duration of the simulation http://www.wildfirelessons.net/uploads/fire_behave_factsheet.pdf
OptFuels	OptFuels integrates existing fire behavior (FlamMap), vegetation simulation (FVS-FFE), and land management planning (MAGIS) tools into one decision support system that supports long-term fuel management decisions. http://www.fs.fed.us/rm/human-dimensions/optfuels/main.php
RANDIG	RANDIG simulates fire spread using the minimum travel time methods and inputs on wind, fuel moisture, and topography.
SVS	The Stand Visualization System (SVS) is a post-processing program for FVS. http://www.fs.fed.us/fmcs/fvs/software/index.shtml
VSmoke	VSmoke is a model that estimates downwind concentrations of particulate matter at 31 fixed distances. http://webcam.srs.fs.fed.us/tools/vsmoke/index.shtml

6. References

- Brown J.K. (1985) The "unnatural fuel buildup" issue. *Symposium and Workshop on Wilderness Fire, Missoula, MT*, J.E. Lotan, B.M. Kilgore, W.C. Fischer, and R.W. Mutch, eds., USDA Forest Service, Intermountain Forest and Range Experiment Station, Ogden, UT, General Technical Report INT-182.
- Calkin D.E., Ager A.A., Gilbertson-Day J., Scott J.H., Finney M.A., Schrader-Patton C., M. Q.T., Strittholt J.R., and D. K.J. (2010) *Wildfire Risk and Hazard: Procedures for the First Approximation*, D.E. Calkin, A.A. Ager, and J. Gilbertson-Day, eds., USDA Forest Service, Rocky Mountain Research Station, Fort Collins, CO (Gen. Tech. Rep. RMRS-GTR-235). Available on the Internet at http://www.fs.fed.us/rm/pubs/rmrs_gtr235.pdf.
- Erl T. (2005) *Service-oriented architecture: concepts, technology, and design*, Prentice Hall PTR, Upper Saddle River, NJ.
- Ferry G.W., Clark R.G., Montgomery R.E., Mutch R.W., Leenhouts W.P., and Zimmerman G.T. (1995) Altered fire regimes within fire-adapted ecosystems. *Our living resources: a report to the nation on the distribution, abundance and health of U.S. plants, animals and ecosystems*, E.T. LaRoe ed., U.S Department of the Interior, National Biological Service, Washington, D.C., 222-224
- Finney M.A. (2002) Fire growth using minimum travel time methods. *Canadian Journal of Forest Research*, **32**, 1420-1424, July. Available on the Internet at http://www.firemodels.org/downloads/flammap/publications/Finney_2002_CJFR_v32_i8_pp1420-1424.pdf.
- Finney M.A. (2006) An overview of FlamMap fire modeling capabilities. In *Fuels management—how to measure success: conference proceedings*, P. L.Andrews and B.W. Butler eds., USDA Forest Service, Rocky Mountain Research Station, Fort Collins, CO, 213-220 (Proceedings RMRS-P-41). Available on the Internet at http://www.fs.fed.us/rm/pubs/rmrs_p041/rmrs_p041_213_220.pdf.
- Forrester E.C. (2007) Technology transition. Presented at the *Software Engineering Institute, Carnegie Mellon University, Pittsburg, PA*.
- Funk T.H., Rauscher M., Raffuse S.M., and Chinkin L.R. (2008) Findings of the current practices and needs assessment for the Interagency Fuels Treatment Decision Support System (IFT-DSS) project. Technical memorandum prepared for the Interagency Fuels Treatment Work Group (IFTWG), by Sonoma Technology, Inc., Petaluma, CA, and the Air Fire Science Team, Seattle, WA, STI-908038.01-3504, December.
- Funk T.H., Corman R.G., Reed J.E., Raffuse S.M., and Wheeler N.J.M. (2009) The Interagency Fuels Treatment Decision Support System (IFT-DSS) software architecture. Software architecture design prepared for the Joint Fire Science Program, Boise, ID, by Sonoma Technology, Inc., Petaluma, CA, STI-908038.04-3565, March.
- Funk T.H. (2010) Interagency Fuels Treatment-Decision Support System proof-of-concept system. Technical memorandum, STI-909029-3876-TM, May.

- Hardy C.C. (2005) Wildland fire hazard and risk: Problems, definitions, and context. *Forest Ecology and Management*, **211**, 73-82.
- Heinsch F.A. and Andrews P.L. (2010) BehavePlus fire modeling system, version 5.0: design and features. General Technical Report by the U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, Fort Collins, CO, RMRS-GTR-249, December. Available on the Internet at http://www.fs.fed.us/rm/pubs/rmrs_gtr249.pdf.
- Joint Fire Science Program (2008) Interagency fuels treatment decision support system conceptual design, v 0.9. Sponsored by the Joint Fire Science Program, the National Interagency Fuels Coordination Group, and the National Interagency Fuels Management Group, August 18.
- Joint Fire Science Program (2009) A powerful new planning environment for fuels managers: the interagency fuels treatment decision support system. *JFSP Fire Science Digest*, December(7).
- Keane R.E., Drury S.A., Karau E., Hessburg P.F., and Reynolds K.M. (2010) A method for mapping fire hazard and risk across multiple scales and its application in fire management. *Ecological Modelling*, **221**, 2-18 (STI-3782, doi:10.1016/j.ecolmodel.2008.10.022).
- Moore G.A. (1991) *Crossing the chasm: marketing and selling technology products to mainstream customers*, Harper Business.
- Mutch R.W. (1994) Fighting fire with fire: a return to ecosystem health. *Journal of Forestry*, **92**(31-33), 31-33.
- National Research Council (1989) *Improving risk communication*, National Academy Press, Washington, D.C. Available on the Internet at <http://www.nap.edu/openbook.php?isbn=0309039436>.
- Newcomer E. and Lomow G. (2005) *Understanding SOA with web services*, Addison-Wesley Professional.
- Ottmar R.D., Burns M.F., Hall J.N., and Hanson A.D. (1993) *Consume Users Guide, Version 1.0*, USDA Forest Service, Pacific Northwest Research Station, Portland, OR (General Technical Report PNW-GTR-304). Available on the Internet at http://www.fs.fed.us/pnw/pubs/pnw_gtr304.pdf.
- Ottmar R.D., Sandberg D.V., Riccardi C.L., and Prichard S.J. (2007) An overview of the fuel characteristic classification system: quantifying, classifying, and creating fuelbeds for resource planning. *Can. J. For. Res.*, **37**, 2383-2393 (doi: 10.1139/X07-077). Available on the Internet at http://www.fs.fed.us/pnw/fera/publications/fulltext/fccs_cjfr/ottmar_etal.pdf.
- Palmquist M.S. (2008) Working summary of the SEI's engagement with the Joint Fire Science Program. Report prepared for the U.S. Department of Defense by the Acquisition Support Program, Software Engineering Institute, Carnegie Mellon University, April.

- Panda D. (2005) An introduction to service-oriented architecture from a Java developer perspective. Available on the Internet at <http://www.onjava.com/pub/a/onjava/2005/01/26/soa-intro.html?page=1>. January 26.
- Peterson D.L., Evers L., Gravenmier R.A., and Eberhardt E. (2007) Analytical and decision support for managing vegetation and fuels: A consumer guide. General technical report prepared by U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station, Corvallis, OR, PNW-GTR-690, January. Available on the Internet at <http://www.wy.blm.gov/fireuse/pubs/AnalyticalDecisionSupport.pdf>.
- Prichard S.L., Ottmar R.D., and Anderson G.K. (2006) *Consume 3.0 User's Guide*, USDA Forest Service, Pacific Northwest Research Station, Seattle, WA Available on the Internet at http://www.fs.fed.us/pnw/fera/research/smoke/consume/consume30_users_guide.pdf.
- Rauscher H.M. (2008) Summary of data related issues as they affect the JFSP IFT-DSS development and deployment. October. Available on the Internet at http://frames.nbii.gov/documents/jfsp/sts_study/ift_dss_task1_data_issues_20081030.pdf.
- Reinhardt E. and Crookston N.L. (2003) *The Fire and Fuels Extension to the Forest Vegetation Simulator*, USDA Forest Service, Rocky Mountain Research Station, Ogden, UT (Gen. Tech. Rep. RMRS-GTR-116). Available on the Internet at http://www.fs.fed.us/rm/pubs/rmrs_gtr116.pdf.
- Sandberg D.V., Riccardi C.L., and Schaaf M.D. (2007) Reformulation of Rothermel's wildland fire behaviour model for heterogeneous fuelbeds. *Can. J. For. Res.*, **37**(12), 2438-2455 (doi: 10.1139/X07-094). Available on the Internet at http://www.nrcresearchpress.com/doi/abs/10.1139/X07-094#.UKq_A4Z2PZA.
- Scott J.H. and Reinhardt E.D. (2001) Assessing crown fire potential by linking models of surface and crown fire behavior. Prepared by the U.S. Department of Agriculture Forest Service, Rocky Mountain Research Station, Fort Collins, CO, Research Paper RMRS-RP-29. Available on the Internet at <http://www.fire.org/downloads/nexus/2.0.0/rp29.pdf>.
- Software Engineering Institute (2008) JSFP program deliverables. Prepared for the Joint Fire Science Program, Boise, ID, January 14, by Carnegie Mellon University, Pittsburgh, PA.
- U.S. Congress (1998) Forest Recovery and Protection Act of 1998, Report on HR 2515, House Report 105–440, part 1, 105th Congress, second session. March 12.
- U.S. Department of Agriculture and U.S. Department of the Interior (2008) *Interagency prescribed fire planning and implementation procedures guide* Available on the Internet at <http://www.nwcg.gov/pms/RxFire/rxfireguide.pdf>.
- Watt S. (2007) Mashups -- The evolution of the SOA, Part 2: situational applications and the mashup ecosystem. Available on the Internet at <http://www.ibm.com/developerworks/webservices/library/ws-soa-mashups2/>. November 8.

Appendix A: Response to Requirements

Table A-1 summarizes the original requirements identified for the IFTDSS and describes how they have been addressed in the implementation of IFTDSS Version 2.0. While it was recognized from the beginning that it might not be feasible to implement all of these requirements in the relatively short development period for IFTDSS Version 2.0, the goal was to meet as many of these requirements as possible.

Table A-1. Summary of the desired requirements for IFTDSS and how they were met in IFTDSS Version 2.0.

Page 1 of 5

Requirement	How Implemented
General Software Requirements	
Modularity.	Each service, software application, data set, and tool has been implemented in such a way that the component may be used independently of other software components.
Extensibility.	The system can be expanded over time to support the incorporation of new tools and data as they become available. As new applications are developed or old ones modified, they can be easily added to system by registering them with the SMF and adding a workflow.
Flexibility.	The system is flexible so users can customize data and model execution to fit their specific project analysis. These customizations can be saved, copied, and shared with other users. As noted under Extensibility, the SMF allows tools, models, and data to be adapted as requirements change.
Portability.	The system is easy to access and use from any standard desktop computer through the Web Application and does not require proprietary software or systems. The core systems have been developed in Java and JavaScript to facilitate portability across platforms and operating systems. However, some of scientific models in the system are native Windows applications.
Ease of use.	The system is straightforward and practical to use through a well-designed interface. Specialized software training or programming skills are not required to run the system and extensive user help and workflow documentation has been integrated into the system.
Maintainability.	The system has a clearly defined structure and well-documented platform (hardware and software) requirements that facilitate maintenance of the system. Further technical documentation was deferred until a host agency is established.

Table A-1. Summary of the desired requirements for IFTDSS and how they were met in IFTDSS Version 2.0.

Page 2 of 5

Requirement	How Implemented
Strategic Requirements	
Development of a unifying software framework to integrate applications.	The IFTDSS software integration framework uses the scientific modeling framework (SMF), a service oriented architecture (SOA), for managing and integrating scientific models and data.
Centralization, organization, and management of fuels treatment data, software models, and analysis tools.	The SMF consists of a core software design and library; a set of scalable service implementations for managing metadata, data, and models; and a suite of tools and support libraries for application and model development.
Development of a registry system for new applications or updates to be made available to the scientific community.	The SMF has defined standards for adding applications (new or updates) to the system. However, no service is currently provided to allow developers to independently register or publish applications.
Development of one or more scientific collaboration tools that can be used by application developers to assist in modifying the various software applications so they function within the new framework.	Although the architecture is capable of supporting model developer collaboration, a service to do this will be addressed when the system has a host agency.
Specification of data standards, which will be supported by all integrated applications.	Data standards have been specified for all currently integrated applications.
Assistance and training for the scientific community, as necessary, to achieve the integration of the various applications.	A Developer's Guide has been written to assist model developers in understanding the system architecture and prepare their models for integration. Training and assistance will be addressed further when the system has a host agency.
Training of fuels treatment specialists, as appropriate, via software user guides, integrated online help pages, and training programs.	User guides, online help, and tutorials have been integrated into the IFTDSS help system. An integrated creation, management, and publication tool was used in developing the help system to ensure maintainability.
Community Requirements	
Software tools to simplify and streamline the process of integrating new models into the IFTDSS.	To be addressed when the system has a host agency.
Technical assistance, including software application programming interface (API) documentation and email or phone support to application developers.	To be addressed further when the system has a host agency.
Easy registration of components, and simplified delivery of applications and updates to users.	To be addressed further when the system has a host agency.
Clear specifications for data standards, and specifications of required APIs that the software applications are expected to support.	To be addressed further when the system has a host agency.

Table A-1. Summary of the desired requirements for IFTDSS and how they were met in IFTDSS Version 2.0.

Page 3 of 5

Requirement	How Implemented
Multiple methods of integrating models into the system.	Models may be integrated into the system directly, through the use of a model wrapper, or by web service calls to an external system.
Technical Requirements	
Allow as many models/modules as possible to work together (including those not yet developed) with a minimum of additional effort required by those application developers to support the framework.	Once models or modules are registered with the SMF, they may be combined by defining the workflow written in XML and adding it to the WebUI Library.
IFTDSS should be able to run from a web browser by any desktop, laptop, or workstation computer connected to the Internet so users do not have to install any software on their local desktop computer.	The IFTDSS Web Application provides unified access to all IFTDSS functionality through a web browser on the user's local computer. However, users who wish to take advantage of optional features, such as viewing results in Google Earth, may need to install other software.
Functional Requirements	
Support the decision support process, analysis steps, and software tools commonly used for fuels treatment planning.	The system supports the decision support process through four guided workflows (hazard analysis, risk assessment, fuels treatment, and prescribed burn planning) and implementation of the Fire and Environmental Research Applications (FERA) Fire and Fuels Application (FFA).
Support visualization of spatial and tabular data, data editing, and user interaction at each processing step.	Tools for visualization and editing of spatial and tabular data are integrated into the system. These tools are provided at each step in the workflows.
Provide data choices.	The system was implemented with standard gridded landscape data, and locally generated data. LANDFIRE spatial landscape data, the National Tree-List Layer (NTLL), and FCCS Fuelbed data are directly accessible without having to import them.
Have data processing and transformation mechanisms to acquire or create and transform input data.	Editing capabilities for spatial landscape and treelist data are integrated into the system. IFTDSS supports conversion of the NTLL to spatial landscape data and the automatic conversion of data units.
Have a quality control, documentation, and audit trail mechanism to support regulatory requirements.	Project analyses are saved as completed, can be archived, and can be shared with other users. The prescribed burn planning workflow leads the user through the completion and production of a burn plan, and has the ability to automatically populate portions of the plan with output from the analyses performed.
Provide guidance (i.e., submodel choices) based on geographic scale and the type of analysis being performed.	In workflows that provide model choices, documentation and guidance is made available through mouse-over help, pull-down help, or the online help system.
Be able to be stopped or started at any processing point.	The user may start a new project or run, perform other functions, or log off, and they will be directed to continue processing at the point where they left off when returning to the original analysis.

Table A-1. Summary of the desired requirements for IFTDSS and how they were met in IFTDSS Version 2.0.

Page 4 of 5

Requirement	How Implemented
Support analytical collaboration.	Fuels treatment analysts can contact other users and publish their methods through the systems collaboration features.
Have a mechanism to perform sensitivity analyses.	Single or multiple input variables can be changed to perform sensitivity analyses. The system supports workflows designed specifically to perform sensitivity analyses by iterating over different input variables.
Recognize user error and explain alternative actions.	The system automatically performs range checks on user inputs and notifies the user if their inputs are outside the acceptable range.
Support scientific collaboration; that is, the system must be able to incorporate new models and tools as they become available through an authorship and publishing mechanism.	To be addressed when the system has a host agency.
Information Technology Requirements	
Must have an operation and maintenance plan and a long-term hosting agency with allocated servers, equipment, and maintenance staff.	To be addressed when the system has a host agency.
Should be fully operational (24/7) and reliable.	To be addressed when the system has a host agency.
Should be designed to function with high-speed Internet.	All user interaction is over the Internet using a standard web browser.
Must support ArcGIS data formats and other commonly-used geographic information system (GIS) data formats.	The system supports Esri shapefiles, landscape (.lcp) data, treelist data, and other structured data used by the models that have been implemented.
Must be designed for inter-operability with other decision support systems in the fire and fuels domain.	The system supports inter-operability with other systems through standard web service connections. This capability is demonstrated through its interaction with the BlueSky Framework.
Performance and Scalability Requirements	
Ability to accommodate up to a maximum of 250 simultaneous users (about 25% of the total user community), running up to 500 computations per hour.	The system makes use of multiple servers to accommodate the required users and computations. Additional servers can be added to support increasing numbers of users and the computational demands of additional models.
Ensure a response time of three seconds or less whenever a user invokes a command in the GUI. In cases of heavy load, or if the user is in the process of running an operation, the system response should be an indication that the command has been recognized by the system and that the server is busy.	With all user interaction through the light-weight WebUI, the system can handle large numbers of users and provides responses in less than three seconds for many analyses. However, the execution of computationally demanding models may require a longer response time; an indication of progress is provided. Ultimately, the number of servers assigned to executing models will need to be optimized once the system has a host agency.

Table A-1. Summary of the desired requirements for IFTDSS and how they were met in IFTDSS Version 2.0.

Page 5 of 5

Requirement	How Implemented
Ensure that if a user repeatedly invokes an operation (because of issues with slow response time, for example) the repeated commands will not interfere with system stability.	Workflows within IFTDSS automatically advance to the next workflow step once an operation is invoked, which prevents redundant commands.
Store all intermediate calculations as they are produced by simulations, and any data the user enters will be stored as soon as it is received. In the case of a server crash, or if the system needs rebooting, all stored data will be available when the user logs back in. In addition, routine server backup processes will be employed to ensure that data are not lost in the case of a server failure.	Workflow progress and intermediate calculations are saved as each step of the workflow is completed. The user is returned to that point in the workflow if the their session is disrupted for any reason. Redundancy and routine backup of data will need to be addressed once the system has a host agency.

Appendix B: Hosting and Security

This appendix describes the current IFTDSS 2.0 hardware and software configuration and security practices. Requirements for hosting IFTDSS 2.0 operationally and potential deployment and operational issues are also discussed.

Current Hardware and Software Configuration

The hardware and software used to support IFTDSS Version 2.0 is summarized in **Table B-1**.

Table B-1. Minimum hardware and software requirements for IFTDSS Version 2.0.

Item	Description
System hardware	Two Windows servers ^a <ul style="list-style-type: none"> • 2 CPUs • 24GB RAM • Intel Xeon 2.27 GHz 16 cores • 1.3TB disk space – hardware RAID 5
System software	Software components ^b <ul style="list-style-type: none"> • Windows 2003 R2 Standard x64 Edition SP2 • Java JRE 6 update 20, 32-bit version • IIS (any version) • Tomcat 6.0.26, 32-bit version Data and database components <ul style="list-style-type: none"> • PostgreSQL Online help components <ul style="list-style-type: none"> • MadCap Flare 8.1

a. Additional servers may be required to support an increased number of users.

b. Some models require that specific libraries be installed on the servers.

Current Information Security Practices

Information security is an integral part of STI's information systems operations. Beyond our need to protect corporate and client information assets, STI supports operation government program, such as EPA's AirNow program, and therefore has implemented systems, policies, and practices needed to meet the requirements of government information security policies. As a government contractor, STI participates in ongoing information security planning, implementation, testing, and assessments. For example, STI operates AirNow under an EPA-approved security plan. Federal Information Security Management Act (FISMA) compliance is assured by meeting NIST requirements, including

- Initial system certifications
- Triennial re-certifications

- Annual security assessments
- Independent Verification and Validation (IV&V)

STI participates in annual security assessments of the AIRNow program and STI's supporting infrastructure through the Automated System Security Evaluation and Remediation Tracking (ASSERT) system. STI's systems and procedures used to ensure information security include

- **Redundant network security gateways/firewalls.** STI uses hardware firewalls to tightly control access to our information systems.
- **Multi-level virus scans.** Scanning for viruses occurs at multiple levels on servers and workstations. Virus definitions are updated nightly and actively pushed to computers in the network, and nightly scans are performed. Real-time protection against viruses is also implemented on servers and workstations.
- **Hardware spam and virus filters.** A real-time hardware spam and virus filter screens all emails before those emails arrive at our email server.
- **Multiple Internet connections.** STI currently maintains four Internet connections, with dynamic routing to provide redundant capability in the event of outages.
- **Backup systems.** STI uses a disk-to-disk-to-tape backup system. Daily backups are made of critical and operational systems, and workstations are backed up weekly. After images of the file systems are created on the backup server, they are copied to magnetic tapes using an automated tape management system. Backup tapes are cataloged, picked up weekly by courier, and stored at a secure offsite facility.
- **Software kits.** These kits contain the latest documentation, instructions, and software needed for restoring data backups and re-deploying our Data Center and Data Management Center.
- **Collocation facility.** STI maintains rack space, communications, and servers at a commercial collocation facility. The facility supports projects requiring redundant servers or high bandwidth, provides an access point for monitoring STI's internal systems from outside our network, and provides an alternate site from which to operate in case of emergencies.
- **Internal and external monitoring systems.** STI operates two systems (one internal and one at our collocation facility) to monitor the state of STI's network and servers. The systems provide tactical summaries of our network and servers, and automated email notification to data management staff in the event of outages, equipment failure, or network intrusions.
- **UPS with email notification of power failures.** Operational systems in STI's Data Center and Data Management Center are powered by two commercial-grade uninterruptible power supplies (UPS) capable of maintaining operation without commercial power. A UPS monitoring system provides web-based access to UPS status and provides email notifications to the data management staff in the event of a power failure.

- **Physical security.** Physical access to the Data Center is gained through a two-factor authentication cipher lock and is restricted to key Information Technology personnel. Building access during business hours is restricted to entry through a front lobby and visitors are escorted. After-hours access requires a physical key and an individual pass-code for the building alarm system, which is monitored remotely.
- **Data Center physical monitoring.** STI's Data Center is monitored for motion (intrusion), temperature, and smoke, with automatic reporting to commercial monitoring centers. These monitoring centers notify key STI staff and police/fire departments as needed.
- **Network scans and security audits.** STI periodically has security audits and network scans performed by external agencies or companies. The results of these scans and audits are used to strengthen our information security practices and close any security holes in our systems.
- **Patch management system.** STI utilizes a patch management system to monitor the status of security patches on all computer systems and push updates to individual computers.
- **Information security training.** STI has an ongoing information security program to train new staff and maintain information security awareness. STI staff members working with government systems are required to undergo annual information security training and testing.

Related Laws, Regulations, and Policies

The following laws, regulations, and policies provide the foundation of our information security plans and practices.

- Federal Information Security Management Act of 2002
- FIPS 199, "Standards for Security Categorization of Federal Information and Information Systems"
- FIPS 200, "Minimum Security Requirements for Federal Information and Information Systems"
- NIST SP 800-37, "Guide for the Security Certification and Accreditation of Federal Information Systems"
- NIST SP 800-53, "Recommended Security Controls for Federal Information Systems"
- NIST SP 800-53A, "Guide for Assessing Security Controls in Federal Information Systems"
- NIST SP 800-60, "Guide for Mapping Types of Information and Information Systems to Security Categories"
- NIST SP 800-18, Revision 1, "Guide for Developing Security Plans for Federal Information"
- OMB A-130, "Appendix III: Security of Federal Automated Information Resources"

Implementation and Operational Hosting Issues

To implement IFTDSS in an operational environment will require considering and resolving a number of issues, including deployment, hardware resources and responsiveness, security, and operations and maintenance. Each of these topics is discussed below.

Deployment

Migration of the IFTDSS to an operational data center will require preparation, training, and assistance. The following items should be considered in the implementation and migration approach. Most of these items may require support from the IFTDSS developer.

- Re-evaluate hardware and software requirements for the operational environment.
- Acquire and configure hardware and software resources.
- Prepare, migrate, and configure the required databases.
- Install and configure the IFTDSS software.
- Verify the installation after implementation.
- Train on IFTDSS management, monitoring, and maintenance.

Hardware Resources and Responsiveness

While the hardware and software being used for IFTDSS are adequate for supporting the current test user group and training session, additional computing resources will be needed to support larger-scale operations. There are about 800 to 1,000 fuels treatment planners (primary users of the IFTDSS) working in various agencies across the United States. It is unlikely that the number of users will increase substantially over time. However, if the IFTDSS is successful, increases should be expected in the number of jobs submitted to the system as the planning process becomes more efficient and more time can be spent performing multiple or scenario-based analyses.

The IFTDSS hosts some CPU-intensive software applications that may take a substantial amount of time to complete (on the order of several minutes to hours in some extreme cases). When CPU-intensive applications are hosted on a back-end web-server, three key issues should be addressed: (1) average and maximum user load, (2) load balancing, and (3) user experience.

To evaluate the risks of a CPU-intensive back-end system, it is necessary to estimate the expected average user load (number of users logged on at one time) and the maximum user load and to determine the amount of CPU resources that will be consumed by a given user. It is important to specify acceptable ranges of response (i.e., how quickly the system will react to a user clicking on a button or link) and performance, or job completion time, for the average system load and for the occasional peak load.

In some cases, IFTDSS users may have a powerful local computer with significant CPU power. However, the web-based architecture forces all system calculations to be handled by

the shared server on the back end of the system. If the server is busy supporting multiple users, user productivity could be reduced by lengthening job completion times (as compared with performing calculations locally). In addition, if users wish to work with data that already reside on their local workstations, those data may need to be transferred to the web server before processing can occur.

Because the IFTDSS is intended to serve many users who may have widely differing local hardware and computing resources, the user experience should be normalized; that is, each user experiences comparable performance regardless of local hardware and computing power. It is generally easier and more cost-effective to upgrade a system's back-end servers to achieve better performance than it is to upgrade a large number of local user computers or workstations.

While the IFTDSS has been designed to be scalable to address these responsiveness issues, it has not been tested with a large number of real users performing the actual analyses they need. Therefore, the actual (real-world) performance characteristics will need to be further assessed once the system is in production and the hardware resources have been scaled to meet those performance needs.

Security

The IFTDSS is a framework that facilitates the integration of new and legacy software applications from the fire and fuels community, thereby streamlining work processes. As an applications framework, the IFTDSS standardizes communications between applications and insulates the applications and their required data sets from the user by a single user interface that interacts with a centralized host for the data and applications. While the IFTDSS does not store, process, or transmit sensitive or confidential information, it will store, process, and transmit data from multiple federal agencies. Therefore, information security must be considered.

While the developers have considered security in the design and implementation of IFTDSS, security issues related to one specific component will need to be addressed: the Authentication Client. A simple, secure authentication client has been implemented in the IFTDSS application for development, testing, and training purposes. This client is a separate service that verifies the credentials and access right of users of IFTDSS. For operational deployment of IFTDSS, a more robust authentication client should be considered. This client may be implemented locally with IFTDSS, or it could be a part of a larger, remote authentication system such as the Fire and Aviation Management Web Applications web site (FAMWEB), which brings together a variety of applications, tools, and services related to interagency fire and aviation management. (FAMWEB is managed by the National Wildfire Coordinating Group (NWCG) and participating agencies.)

As the IFTDSS becomes more stable and operational in the fuels treatment planning environment, the hosting agency will need to implement, test, and document security controls required under federal information processing standards. When an agency has been identified for hosting the IFTDSS operationally, that agency's information security staff should develop a System Security Plan (SSP) and ensure that appropriate security requirements and accessibility

requirements are met. Note that a draft SSP was prepared early in the development process that could provide a foundation for the final plan. The final plan will need to be revised to reflect the current system and requirements of the hosting agency.

Operations and Maintenance

The hosting organization will need to provide staff for ongoing operations and maintenance (O&M) that would include the following tasks:

- Routine support: assisting users with setting up accounts, addressing technical questions, and providing assistance to users who may be having difficulty using the system.
- Monitoring system health and performance.
- Maintaining software, which includes updates to the IFTDSS software and managing patches.
- Receiving, testing, and deploying updates to the IFTDSS software and databases.
- Receiving, tracking, and resolving IFTDSS software issues.
- Receiving, tracking, and reporting enhancement requests.
- Training users as needed.